NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

HUMAN BEHAVIOR REPRESENTATION OF MILITARY TEAMWORK

by

Jon E. Ellis
Michael W. Martin

June 2006

Thesis Advisor:	Christian Darken
Co-Advisor:	Jeffrey Crowson
Second Reader:	Leroy A. Jackson

This thesis done in cooperation with the MOVES Institute.

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** June 2006	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE: Human Behavior Representation of Military Teamwork			**5. FUNDING NUMBERS**
6. AUTHOR(S) Jon E. Ellis, Michael W. Martin			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) U.S. Army Training and Analysis Center – Monterey, Monterey, CA 93943			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited			**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

This work presents a conceptual structure for the behaviors of artificial intelligence agents, with emphasis on creating teamwork through individual behaviors. The goal is to set up a framework which enables teams of simulation agents to behave more realistically. Better team behavior can lend a higher fidelity of human behavior representation in a simulation, as well as provide opportunities to experiment with the factors that create teamwork. The framework divides agent behaviors into three categories: leadership, individual, and team-enabling. Leadership behaviors consist of planning, decision-making, and delegating. Individual behaviors consist of moving, shooting, environment-monitoring, and self-monitoring. Team-enabling behaviors consist of communicating, synchronizing actions, and team member monitoring. These team-enabling behaviors augment the leadership and individual behaviors at all phases of an agent's thought process, and create aggregate team behavior that is a hybrid of emergent and hierarchical teamwork. The net effect creates, for each agent, options and courses of action which are sub-optimal from the individual agent's standpoint, but which leverage the power of the team to accomplish objectives. The individual behaviors synergistically combine to create teamwork, allowing a group of agents to act in such a manner that their overall effectiveness is greater than the sum of their individual contributions.

14. SUBJECT TERMS Human Behavior Representation (HBR), Agent-based Modeling, Artificial Intelligence, Teamwork, Team-enabling Behaviors			**15. NUMBER OF PAGES** 93
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

HUMAN BEHAVIOR REPRESENTATION OF MILITARY TEAMWORK

Jon E. Ellis
Major, United States Army
B.S., United States Military Academy, 1993

Michael W. Martin
Captain, United States Army
B.S., United States Military Academy, 1997

Submitted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE IN MODELING,
VIRTUAL ENVIRONEMNTS AND SIMULATION (MOVES)**

from the

**NAVAL POSTGRADUATE SCHOOL
June 2006**

Author:	Jon E. Ellis Michael W. Martin
Approved by:	Christian Darken Thesis Advisor
	J. Jeffrey Crowson Co-Advisor
	Leroy A. Jackson Second Reader
	Rudolph Darken Chairman, MOVES Academic Committee

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This work presents a conceptual model for the behaviors of artificial intelligence agents, with emphasis on creating teamwork through individual behaviors. The goal is to set up a framework which enables teams of simulation agents to behave more realistically. Better teamwork can lend a higher fidelity of human behavior representation in a simulation, as well as provide opportunities to experiment with the factors that create teamwork. Our framework divides agent behaviors into three categories: leadership, individual, and team-enabling. Leadership behaviors consist of planning, decision-making, and delegating. Individual behaviors consist of moving, shooting, environment-monitoring, and self-monitoring. Team-enabling behaviors consist of communicating, synchronizing actions, and team member monitoring. These team-enabling behaviors augment the leadership and individual behaviors at all phases of an agent's thought process, and create aggregate team behavior bridges the gap between emergent and hierarchical teamwork. The individual behaviors synergistically combine to create teamwork, allowing a group of agents to act in such a manner that their overall effectiveness is greater than the sum of their individual contributions.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

We would like to thank our families for their continued understanding and steadfast support as we plugged through this arduous, time-consuming research process. We know that it is not easy raising a family in the military and it is even tougher when dad has to write code until all hours of the night. We absolutely could not have finished this journey without you.

Next, we would like to thank our dedicated advisors for allowing us to work through a proof-of-principle thesis. As a result of our work, we certainly better understand the complexities of simulation development and feel that we are better prepared should we ever find ourselves in a management role in the simulation community.

For Professor Darken, thanks for your flexibility during our development phase. You allowed us to more or less feel our way through the development process. From this we learned: first, that this may not always be the best technique and second, that it is very easy from a student's perspective to bite off more than we can chew.

For Professor Crowson, thanks for the support early in the model conception phase of our research. The teamwork class that you sponsored for us was absolutely instrumental in the development of our abstract model. It was always evident that you support the student, above all else, during your teaching here at NPS. Thank you.

For Mr. Jack Jackson, thanks for giving us a "gut-check" from time to time. You kept us on path realistically with what is viable in the simulation world of today. You also provided a senior-level look at our research. Your feedback was always spot-on.

To conclude, we would be remiss if we did not thank Director Rudy Darken for his support and the support of his superb staff and faculty within the MOVES Institute. We would not have been able to work through many of our development issues without the professional and educational support of your team.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION AND MOTIVATION

A. PROBLEM STATEMENT

In many current military simulations, including those with entity level fidelity, the modeling of soldiers does little to represent the synergistic effects of teamwork on the battlefield. In reality, the effectiveness of teamwork within any military unit is arguably the most important feature of that team. More than numbers or equipment, how well the elements of a unit work together is critical role to the success or failure of that unit in reaching its objective. As teamwork is a fairly intangible feature, however, it is hard to represent it either with quantitative or qualitative measurements. As a result, models and simulations overlook this vital aspect, represent it implicitly at best, or assume it away. While this assumption is acceptable for many applications, in certain simulations, the result is unrealistic behavior that degrades the quality of the overall simulation. For our analysis, we will focus on entity level representations of small teams, as applied in the context of tactical training simulations. Our aim is to present an efficient way of making the representation of Soldiers better, specifically by improving the modeling of teamwork.

B. MOTIVATION

1. Credible Soldier Human Behavior Representation for Simulation

From constructive simulations at Corps and Division levels, to Platoon sized engagements in the virtual simulators, human behavior representation of simulated entities is an increasingly important part of modern training. At all echelons, the limitations of human behavior representation are compensated for by increased operator involvement in driving the training behind the scenes. By developing soldier representations which can react realistically in a team, we present opportunities for soldiers in training situations to hone their combat skills against an enemy who will react realistically to the evolving battlefield situation without requiring direct operator input.

The increased realism could improve both the quality of the training value and the immersion of the soldiers in the simulation, as well as providing more realistic results from analytical simulations.

2. Representation of Leadership

In order to develop effective teamwork, we must also develop at least a basic representation of leadership. The lack of leadership representation within simulated soldiers creates a requirement for constant human micromanagement of all but the most rudimentary of desired behaviors by simulated soldiers. Rather than controlling a unit, the operator must individually control each entity to achieve maximum realism. The effect on training is that enemy forces must be micromanaged to a degree where they are no longer autonomous. This increases the personnel resource requirement for training, preventing simulations from achieving their full potential. Implementing a representation of AI leadership within computer controlled forces could reduce the common micromanagement burden associated with AI or SAF (Semi Automated Forces) forces.

3. Team Size Effects

One of the most basic limitations of teamwork is the ability of the leader to manage his subordinates. Current military doctrines and research indicate that a leader is most effective with between three and five subordinates (Pew, 1998). There are several reasons for this, but at the core is the complexity of combat tasks versus the ability of one entity (a leader), to manage different activities. As a result, the degree of control or unity within a unit is inversely proportional to the number of soldiers or sub entities within that element. As the number increases, the effectiveness of the teamwork decreases. A single entity might be the only “perfect” team as it is completely harmonious with itself, thus displaying the perfect teamwork of one, as long as the entity is rational and moderately skilled.

The natural compliment to this inverse relationship between teamwork and size of a unit is the raw power of the unit. This is the aspect of teams that is most commonly captured in simulations. The larger a unit is, the more firepower it has, and the more casualties it can sustain. This basic advantage of size will never change. However, with our work, we hope to balance the size advantage with the unwieldy nature of large units controlled by a single leader, as opposed to a hierarchy of leadership. The interplay of factors creates a theoretical bell-shaped relationship between unit size and effectiveness. As the size increases, the effectiveness increases to a maxima and then decreases as the size becomes a hindrance to effective action.

4. Model the Translation of Situational Awareness into Team Behavior

One specific root cause of the inherent obstacles to creating team harmony is the inability of one person to perfectly convey his own situational awareness to those around him. It is the role of the leader within a group to collaborate with the team members to create a shared understanding of the events surrounding him, melding his own perception with the composite perceptions imperfectly conveyed to him by his team, and then communicating his vision and acting and directing action based on that his understanding. Implicit in the role of all team members is to communicate relevant situational awareness gained so that the team has a shared view of their world.

To an extent, this offloading of processing responsibility to the team leader makes for more elegant resource usage of the team in simulations. The mental processing time (CPU usage in a simulation) is reduced for team members not in leadership positions. This also reflects the reality of team dynamics, where team members typically focus on their own responsibility and react to directives from their leadership.

The end goal for our implementation is a realistic team which models teamwork in an efficient and believable manner. This will aid to validate our model and possibly encourage future implementations of our abstract model.

C. LIMITATIONS

It is important to note that our model does not include a strict representation of human leadership. Our work tries to distill some fundamental aspects of human leadership, in an effort to more accurately model it. By no means is our leadership model an exact replica of human leadership, with all its intricacies and interpersonal dynamics. However, we will try to model will represent teamwork implicitly, rather than explicitly. We will not allow our simulated soldiers to act with one mind, in perfect accordance. Though a “shared mind” would be the epitome of teamwork, and something that human teams strive for, this is not realistic behavior, and would result in rather frustrating training. Our goal is to capture the spirit of interpersonal leadership as it affects teamwork, rather than to model the mechanism it perfectly.

To that end, we have found difficulty in finding a simulation which we can easily manipulate to reflect our theoretical model. As a result, we have built a simulation from the ground up, allowing us to control the way in which the agent operates and what affect the agent’s processes. The simulation lack some of the complexity of more mature simulations, but allows us to focus on the aspects we identify as being relevant to our model. It will consist of a simple representation of the agents and a very minimal environment, consisting of primarily other agents, weapons fire, and communications between agents.

II. BACKGROUND AND RELATED WORK

A. INTRODUCTION

The Soldier Modeling and Analysis Working Group (MAWG), conducting research at the TRADOC Analysis Center-White Sands Missile Range (TRAC-WSMR), issued a Technical Report dated 31 March 2004, identifying several shortfalls in existing models and simulations to effectively model the soldier and small units. The majority of these capability gaps are in modeling Situation Awareness, Decision Making, and Teamwork of both individual soldiers and small teams. In order to move forward into a deeper discussion of our model and research, we will first provide some background information to help define the context of the problem.

B. HUMAN BEHAVIOR REPRESENTATION

Human Behavior Representation (HBR) continues to be an expanding field in the Modeling and Simulation community, especially within the Department of Defense. The reasons for this are numerous, but within the context of this paper the push for realistic HBR modeling exists mainly because of the community's desire to understand and represent how leaders make decisions and how teams operate effectively. The military objective of such focus is to better explore and better train the situations and missions soldiers will encounter in military operations where, despite technology advances, the human element remains preeminent. The National Research Council's Commission on Behavioral and Social Sciences and Education, clearly states the importance of research in the area:

> The modeling of cognition and action by individuals and groups is quite possibly the most difficult task humans have yet undertaken. Developments in this area are still in their infancy….Human behavior representation is critical for the military services as they expand their reliance on the outputs from models and simulations for their activities in management, decision making, and training. (Pew and Mavor, 1998)

While technologies have certainly advanced in the past few years, current simulations often fail to model human behavior to the level of fidelity required for reliable, autonomous military simulations.

C. AGENT-BASED MODELING

The heart of teamwork dynamics is the complex interpersonal relationships which inherently govern group actions. This thesis will approach these relationships through the framework of agent-based modeling. In addition to allowing for a high degree of fidelity within the behaviors of the agents, agent-based modeling also provides the potential for emergent behavior which could prove insightful into teamwork, and which is perhaps even a qualifier of true team behavior. Emergent behavior is generally described as a control technique that does not utilize explicit environmental models (Russell and Norvig, 2003, pg 931), in contrast with control techniques that decide agent behavior based on complex internal models of the agent's environment. Programs such as MANA, PYTHAGORAS, and ISAAC are commonly used to gain insights into human behaviors in conflict situations. However, these agents tend to be purely reactive, applying weighting rules and "attractions" to various events and entities within their world. True military teamwork behavior may also be governed, by and large, by such rules, but must also be projected against a more informed cognitive foundation. This foundation supports more sophisticated representation of the agents' perception's and interactions in order to shape each individuals behavior.

With the increase in specificity, which is implicit in creating a cognitive foundation for agent behavior, the domain for the scope of this work is narrowed significantly by placing it within the military framework. The target agents, for this thesis, will be confined to purely military behaviors, which, at their root, can be boiled down to the mantra "shoot, move, communicate." It is these three actions which will form the basis for all other behaviors for the agents.

D. SITUATION AWARENESS

In making decisions, a human, in our case a soldier, uses mental processes which ultimately provide the individual with one or more possible courses of action. When the soldier acts, a decision has been made. A critical component of the decision-making process is the act of maintaining pertinent information about the current situation. Volumes have been written about this process and there are a plethora of terms which are used to describe it. The United States Army calls it Situation Understanding (SU). There is still an ongoing debate over the most accurate phraseology. We will refer to it here as Situation Awareness (SA). The most widely-accepted and general definition of SA is, "the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning and the projection of their status in the near future" (Endsley, 1988). This definition is fully consistent with emerging Army doctrine. In general, SA is the individual's mental model of the situation. The individual modifies that mental model as a situation unfolds and he gathers information, then the individual makes decisions as required based on this mental model.

1. Sensation, Perception, and Cognition

In general, SA is obtained through a process of sensation, perception and cognition. During the sensing phase, raw data is obtained primarily through the use of an individual's five senses. Whether or not an individual extracts certain data is largely dependent upon their selective attention during sensing. The data is then processed by an individual's long-term memory, prior knowledge, through the top-down process of perception. Within this process, the sensed data becomes information. This information is then fed through an individual's cognitive process where experience and situational information are applied in an individual's working memory. It is during this stage that rehearsing, planning, understanding, visualizing, decision making and problem solving combine to generate actionable information (Wickens, 2004). This actionable information is called Knowledge.

2. SA in Teams

Extending the concept of SA to teams is a bit more complicated and very often controversial. There are some key concepts which are useful in modeling teamwork. The concept of shared mental models is at the root of the discussion of SA in teams. Obviously, it is impossible for the members of a team to actually share a mind. It is, however, possible to model teamwork in such a manner. In fact, to create a separate entity which represents the team SA and decision making capability is very common. While it is arguable whether or not this is the best approach, it is important to understand exactly what is being modeled in this technique. The concept of shared SA is essentially the intersection of all the team members' SA. Thus, shared SA is a shared understanding of that subset of information that is necessary to meet each individual's goals (Endsley, 2003). Implicit in the phenomenon of shared SA is, "knowledge of the status of other team member's tasks to the degree that they impact on one's own tasks and goals" (Endsley, 2003).

E. DECISION MAKING

For this research, we will focus our discussion of decision making on the realm of Naturalistic Decision Making (NDM) since much of the modeled decision making behavior of small teams at the tactical level is accomplished through recognition. A simple definition of NDM is the way people use their experience to make decisions in field settings (Zsambok, 1997). In 1993, Orasanu and Connolly identified eight defining characteristics of the environment in which real-world decisions are made (Orasanu and Connolly, 1993):

- Ill-Structured problems
- Uncertain, dynamic environments
- Shifting, ill-defined, or competing goals
- Action/Feedback loops
- Time stress
- High stakes

- Multiple players
- Organizational goals and norms

These factors certainly describe very well the ever-changing, dynamic environment in which our Nation's military forces conduct the range of military operations. It is within this very environment that every soldier and leader must operate for the unforeseeable future.

1. Expertise

The initial thrust of research within the realm of NDM was led by Dr. Gary Klein. His widely-accepted model of Recognition-Primed Decision Making (RPD) is at the forefront of most research in the arena of Human Behavior Representation (HBR). Central to development of his model is the study of how experienced and skilled individuals or experts make decisions. Individual expertise of team members is very often studied as a measure of how well a team performs a certain task or mission. Obviously, knowing where to focus one's attention, or knowing what to ignore in a given situation are characteristics of an individual with some experience. While research suggests that there are clear distinctions between a novice and an expert in any given domain, the NDM paradigm usually applies to any decision maker in a decision situation where that decision maker has some applicable experience. Take a military officer for instance; an officer rarely remains in a single duty position more than two years and even if an officer serves in a similar capacity as he or she advances in rank, the increased duties and responsibilities create a very different dynamic and on-the-job learning is always occurring. So, in the case of the military officer, it is fairly safe to say that he or she is rarely an expert in any one field. However, throughout the course of a military career, officers are constantly faced with uncertain, time-critical, high-stress, dynamic environments in which high-stakes decisions must be made. In this regard, one may be justified in labeling a military officer an expert decision maker in relevant military situations with the caveat that expertise in this context is about having sufficient relevant training and experience to operate beyond the novice level rather than about displaying the highest level of proficiency. In the Army, we call these experts leaders and every team has one.

2. Leadership

The simple definition of NDM is the way leaders use their experience to make decisions in field settings. In this regard, leaders do not necessarily have to be experts in any one field. If leaders are to be characterized as experts in anything, it should be in their ability to make decisions in environments similar to those outlined by Orasanu and Connolly (Orasanu and Connolly, 1993).

F. COHESION AND TEAMWORK

The concept of a team and of teamwork is at the very core of the Army. Without a team, the Army would not exist. And without teamwork, the Army could not function. Borrowing from Army recruiting commercials, an Army of One is made only possible in an Army of many acting as one through teamwork. Implicit in these commercials are the very essence of the Army's structure and the goal of every unit leader, a cohesive team that blends teamwork with individual initiative.

1. Cohesion

Cohesion is the sinew which holds a team together. Without cohesion, a team would be ineffectual and resemble, instead, merely a group of individuals. The concept of cohesion has several defining characteristics designed to de-emphasize individualism within the soldier (Henderson, 1985):

- Cohesion is inversely proportional to the size of the unit with three to nine soldiers being the optimal lowest unit size.
- Outside threats perceived by the group cause it to coalesce and pull together to face the common danger.

- Casualties can significantly weaken group cohesion, especially casualties that are considered "wasteful" by soldiers….
- A cohesive unit provides the major source of esteem and recognition for unit members.
- An observation-and-reporting system that is self-correcting for deviance from group norms by mobilizing peer groups or leadership presence in order to correct individual behavior is also necessary.

Observers of men in combat have called attention, "again and again to the fact that the most significant persons for the combat soldier are the men who fight by his side and share with him the ordeal of trying to survive" (Henderson, 1985). S.L.A. Marshall, in his book *Men Against Fire*, further described this phenomenon of battlefield courage and the resulting unit cohesion as the byproducts of the mere fact that the individual fighting man did not want to let the man next to him down. This is why he had to hold the line, not simply because that was the given mission. It is the existence of the team and the bonds that formed through training and fighting together which foster teamwork and the ability to accomplish the mission (Marshall, 1978).

2. Teamwork

“[T]he primary function of the organization is to provide purpose to the cohesive unit in the form of goals and objectives” (Henderson, 1985). In the military, it is the leader who provides this function for the team. It is in executing these goals and objectives that the team exhibits teamwork. Salas, Dickinson, Converse, and Tannenbaum (1992) define a team as: “…a distinguishable set of two or more people who interact dynamically, interdependently, and adaptively toward a common and valued goal/objective/mission….” In her book, *Designing for Situation Awareness*, Mica Endsley further defines the required characteristics of a team as: the existence of a shared common goal, team members have predefined roles, and these individual roles or duties are interdependent. It is the dynamic interactions of these interdependent roles that often times is most interesting in the study of teamwork and teams.

3. Expert Team vs. Team of Experts

During our discussion of our motivations and interest in pursuing this research, we briefly discussed a comparison of an expert team with a team of experts. To determine which team will statistically and consistently perform the best is a very interesting research topic, but not a goal of our research. We are, however, interested in showing that, in our view, an expert team is far more adaptive to changing circumstances not uncommon in a fast-paced, ever-changing environment such as those experienced in military operations in urban terrain. Further, it is also our belief that an expert team will always exhibit better teamwork. Again, this is not to say that an expert team will perform better, just that it will work together better as a team in the accomplishment of its assigned mission. It is important to note that both types of teams may effectively accomplish their assigned missions; but, that expert teams will better leverage the contributions of its individuals in achieving the team goals.

G. ARTIFICIAL INTELLIGENCE TEAMWORK IMPLEMENTATIONS

There are many techniques in use today for representing teamwork in Artificial Intelligence (AI) implementations. Two popular methods are at opposite ends of the teamwork modeling spectrum. They are the decentralized and centralized, or squad-AI, approaches.

1. Decentralized Modeling Approach

The decentralized approach is so called because of the designed absence of a leader. All agents have the same behaviors and the interactions between these individual agents determine the overall team behavior (van der Sterren, 2002). Another important characteristic of the decentralized approach is the concept of emergent behavior.

The phenomena of emergent behavior, often referred to as self-organization, is the central theme in a decentralized team AI modeling approach. In his book, Emergence, John Holland provides this definition:

> Emergence is above all a product of coupled, context-dependent interactions. Technically these interactions, and the resulting system, are nonlinear: The behavior of the overall system cannot be obtained by summing the behaviors of its constituent parts. We can no more truly understand strategies in a game board by compiling statistics of the movements of its pieces then we can understand the behavior of an ant colony in terms of averages. Under these conditions, the whole is indeed more than the sum of it parts. However, we can reduce the behavior of the whole to the lawful behavior of its parts, if we take the nonlinear interactions into account. (Holland, 1998)

Stuart Kauffman (1994), in *At Home in the Universe*, provides an additional explanation that the parts of a system create a macro-level organization that is not due to any directed action on the part of any internal or external factor (Koehler, 1994). Thus it is through emergence that the actions of each team member agent and their corresponding interactions result in team behavior, or teamwork, in the absence of a leader. While this is a popular method for modeling teamwork, it is imp0rtant to note that it is nearly impossible for autonomous agents to reach a unanimous decision and execute that decision in a manner representative of coordinated behavior, or teamwork. Thus, from this perspective, a decentralized AI team lacks autonomy.

2. Centralized Modeling Approach

A popular method for solving the lack of autonomous teamwork, described above, is to take a more centralized approach where a leader provides focus for the team. This is often accomplished through the use of a separate squad-level AI, or “hive-mind”, where an additional agent represents the collective mind of the team. This squad-level AI collects all the information from the environment, to include reports provided from the individual team member agents, performs a sort of cognitive assessment of the situation, and makes a decision for the entire team. It then directs the individual agents accordingly in a seemingly coordinated team effort. This method is built on the concept discussed above of a shared mental model. An important distinction to make here is that the squad-level AI is not a member of the team. Thus, while the team may be able to operate autonomously, it is not quickly able to react to unexpected scenarios (van der Sterren, 2002). For example, by centralizing the decision-making, individual team members may not be permitted to react in situations which, by proximity, may be more important to the individual than to the entire team. Additionally, this approach often does not allow for a detailed analysis of the individual team member behaviors and their interactions; since, all decisions and actions are essentially being guided by an agent outside of the team.

H. URBAN OPERATIONS

Urban operations span the range of military operations, from Major Combat Operations to Homeland Defense & Civil Support. Each of these separate missions is dynamic and complex in their own right; however, it is entirely possible that within the limits of a large city, a single unit could execute any number of these very distinct missions concurrently. Marine Corps General Charles Krulak offers the following description:

> The Corps has described such amorphous conflicts as -- *the three block war* -- contingencies in which Marines may be confronted by the entire spectrum of tactical challenges in the span of a few hours and within the space of three contiguous city blocks.

For this very reason, the U.S. Army's current training doctrine is that every unit must be prepared to quickly and seamlessly transition between any of the range of military operations. With an ever shrinking force, this requirement presents many potential training challenges which must be addressed.

In setting the stage for such training, it is critical to sufficiently flesh out the operational environment. United States Army Major General Robert H. Scales, Jr., had this to say of conducting operations in such environments:

> Urban warfare, fighting in cities, war in 'complex terrain.' To the casual observer, the words seem detached, almost pristine. However, the words are strikingly real to military professionals who have seen the images of great destruction and excessive casualties in cities such as Berlin,

> Stalingrad, Hue, and Beirut. Urban warfare, a subject that many military professionals would prefer to avoid, is still with us. Moreover, it may be the preferred approach of future opponents.

According to the Joint Publication on Joint Urban Operations, all urban areas share three main interrelated characteristics: a complex manmade physical terrain, a population of significant size and density, and an infrastructure that supports the population and perhaps the region or nation. These characteristics comprise the urban battlespace. It is this battlespace which must first be understood prior to the executing urban operations. This requirement lends itself to gaining and maintaining battlespace awareness.

Within the context of this battlespace, it is extremely difficult to conduct realistic and synergistic training. Live training would certainly allow for the human factor to be explored to some degree; however, it is nearly impossible to recreate the realistic battlespace, especially the population. Virtual training or simulation could offer some insight into planning and command and control of urban operations; however, the challenges of level of immersion and fidelity always exist in these already very costly simulations. Finally, Constructive simulation offers an environment where the entire urban operations battlespace could be modeled to the desired level of fidelity on a personal computer. Obviously, this would not be cost prohibitive, but how would one model the soldiers, the team and the teamwork?

This brings us to the problem at hand, a model of realistic Human Behavior Representation. The ability to decipher how a team effectively gains and maintains battlespace awareness, makes decisions and executes team-tasks in such a complex and dynamic environment is paramount to the future success of our forces.

THIS PAGE INTENTIONALLY LEFT BLANK

III. TEAMWORK MODEL

A fundamental assumption for anyone who advocates for or uses “teams” is that by organizing individuals into teams, the team becomes greater than the sum of its parts. However, in many current examples of artificial intelligence representing teams, especially those in tactical military settings, such benefits are rare. In fact, frequently AI teams tend to behave as less than the sum of their parts, or at least highlight the short falls of AI. Some of these shortfalls include: agents blocking the field-of-view of other team members, agents following the exact same route to a destination or target, and agents being oblivious to nearby threats while their teammates are taking fire (Orkin, 2003). This work presents a paradigm for teamwork as modeled by agents. The intent is to highlight critical and often missing or under-emphasized behaviors which are necessary to replicate the multiplicative benefits of teamwork, and in doing so, create a behavioral structure for individuals in team organizations. Additionally, this work examines the teamwork phenomena from the bottom up, rather than top down perspective, in an attempt to understand how individuals and their independent actions create a synergistic effect upon the performance of the team as a whole.

We will use the phrase “team-enabling” to denote the focus of our model: those specific behaviors by the team member that create synergistic teamwork. The inclusion of the word ‘enabling’ is used to avoid confusion with the idea of “team behavior”, which is often used to describe the aggregate behavior of all members in a team. In general, we will consciously avoid the aggregate concept of “team behavior”, along with other phrases such as “collective knowledge” or “team cognition” which lend themselves to viewing the collective team as some sort of entity which has its own set of properties.

The U. S. Army’s OneSAF system takes that approach, and creates non-physical team entities outside the individual team members with properties and processes beyond those contained in the individual members. This “hive-mind” implementation, or similar “squad-level” AI solutions from the entertainment industry, advocate this disembodied leadership structure, where additional non-physical agents represents the “leader” at various levels and all inputs are assessed by these “leader” minds, which in turn make decisions given the current situation, and issue guidance to the individual agent team members (van der Sterren, 2002 and Reynolds, 2002). Other solutions stop short of employing non-physical team entities to control behavior, utilizing, instead, informational blackboard tools through which agents can reserve paths and send information signals to each other in order to create the illusion of teamwork (Orkin, 2003). On the other end of the spectrum is the decentralized approach which emphasizes emergent "team-like" behavior. Van der Sterren's discussion of emergent behavior resulting from the communication between agents of 'intent and observations' demonstrates the potential for creating emergent teamwork behavior, but by his own admission, can lead to a decentralized group incapable of decisive action (van der Sterren, 2002). While this technique does focus on individual behaviors and attempts to highlight the application of the behaviors which we have categorized as "team-enabling", it does not seem to accord them the central focus of teamwork discussions. Our model places this emergent behavior upon a backbone of hierarchy, capitalizing on this resulting teamwork while still creating effective, centralized teams who can take unified action when needed.

While the above techniques may be elegant solutions to teamwork, for our purposes, we believe them to be oversimplifications or "work-arounds" that obscure the true nature of teamwork and the interactions of individual behaviors, which constitute teamwork. Our model is predicated on the notion that the only properties that a team possesses are its individual members, and that the dynamics which generate teamwork are nothing more or less than the results of behaviors of the interacting members of the team. Team dynamics are simply the sum of individual behaviors. However, from these behaviors, synergistic interactions result that give a team the effect of being more than the sum of its parts. By examining teamwork from the perspective of individual agents, we are better able to appreciate the manner in which individual behaviors interact, which is an aspect that is easily lost when looking at the phenomena of teamwork as a whole.

The primary focus of this work is on small, military teams. Specifically, the main consideration is of a dismounted, infantry-based, fire team, consisting generally of four soldiers. Limiting our consideration to this specific kind of team helps scope the problem and avoid confounding factors in the discussion of teamwork. The military fire team has predefined roles. There is one team member who is permanently the "team leader", and all other team members are simply that, “team members.” Unlike other types of teams, leadership is not open to negotiation, nor is it likely to shift between the members of a team (discounting the effects of casualties). The military context also allows us to simplify the range of inputs and outputs available to each agent. We can limit the inputs to the perception of objects, such as other agents (friendly and enemy), terrain, weapons fire and communications. For outputs, the available range of actions for each agent can be limited to the military mantra of “move, shoot, and communicate”. Additionally, the arena of military team AI is ripe for discussion as there is a wealth of examples, both military projects and the entertainment industry of AI agents who to one degree or another, model teams.

Another important limitation of this work is that it will not focus on the techniques of leadership that guide a team, or even how effective leadership may be. Instead, it will focus on the role of the “team members”, and how they behave in order to become part of a fully functioning team. This by no means is intended to diminish the importance of leadership, or imply in any way that leadership is not a necessary and critical ingredient of “teamwork”. Rather, this is an acknowledgement that the topic of effective leadership and its psychological components, such as loyalty and motivation, are beyond the scope of this work.

Indeed, it is our intent to assume away many of the psychological and emotional factors of leadership and teamwork by looking at teams from the perspective of an AI agent. This allows us to focus on the mechanics of teamwork and forgo additional confounding factors of psychology and ego. By our design, the agents in our teams will always be considered to be rational automatons, which make decisions without interpersonal emotional bias. As a result, our team members will never disobey because he does not “like” his team leader, for example. In minimizing this emotional/social aspect of agent interaction, our agents will all be exactly alike, each with equal leadership and follower potential, who merely take their designated roles by random chance.

As we look at the behavior of individuals within the teamwork context, our model divides the agent’s behaviors into three categories: “leadership” behaviors, “individual” behaviors, and “team-enabling” behaviors (See Figure 1). These behaviors are a combination of external inputs, internal processes and external actions or outputs.

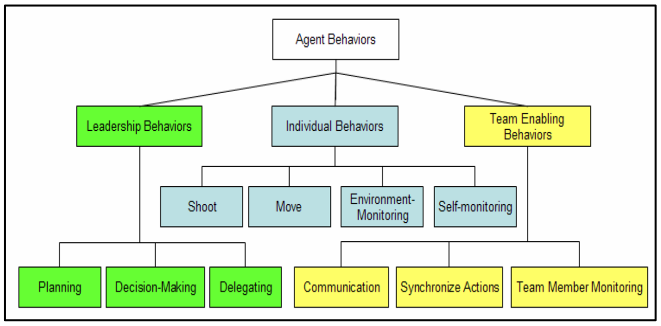


FIGURE 1. BEHAVIOR HIERARCHY

We define “leadership” behaviors as the decision-making behaviors by which an agent chooses which outputs or actions to execute. The three “leadership” behaviors are Planning, Decision Making, and Delegating. All agents, for our purposes, conduct “leadership” behaviors, with the only difference being that only leaders require the Delegating behavior. Additionally, this paradigm also allows for our theory of teamwork to be expanded into a larger hierarchy of leadership, in a “team of teams” concept. In military context, this would equate to higher levels of organization, such as a platoon. A generic platoon structure might have one platoon leader with three subordinate squad leaders. Each squad leader has two fire team leaders, and each fire team leader has three subordinate soldiers (see Figure 2).

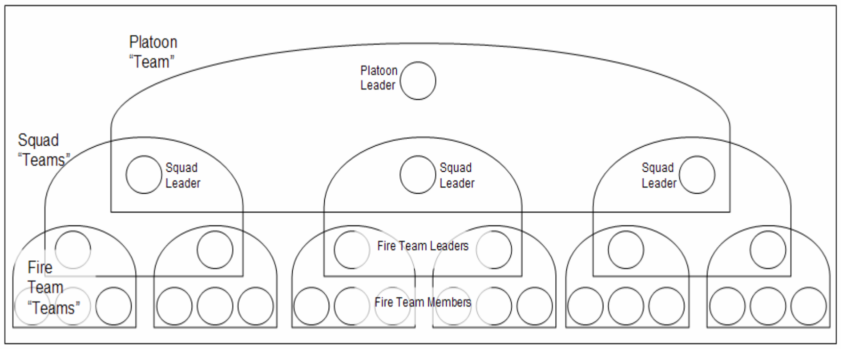


FIGURE 2. PLATOON DIAGRAM OF TEAM OF TEAMS

If the platoon leader is considered the leader of the platoon “team”, then his squad leaders may be considered the team members, for the conceptual team at the platoon level. In turn, the squad leaders are themselves leaders for the conceptual teams at the squad level. In these squad “teams”, the fire team leaders are the subordinate team members. And finally, at the fire team level, the fire team leaders are the “team” leaders, and the lowest ranking soldiers are the “team” members. The teamwork processes which govern member’s behaviors and interactions with each team are the same, even though many of the agents are likely to be both leaders and subordinates in overlapping teams. Agents receive input from either their subordinates or their own leaders, reenter a planning and decision making process, and then delegate the changes to their subordinates as appropriate. The only difference between a leader (at any level), and the lowest subordinate (who is not a leader) is that the lowest subordinate delegates his plan only to himself.

“Individual” behaviors are the simple behaviors which can be executed by the individual agent; specifically, Moving, Shooting, Environment Monitoring, and Self Monitoring. Environment Monitoring is the collection of behaviors which provide information about the environment to the agent, such as seeing or hearing. Self Monitoring is a subtle but important variation on Environment Monitoring, in that the agent assesses its own status in order to provide pertinent information to its decision making process. As an example, an agent might evaluate its health or its estimated proximity to completing a task. Being the simplest behaviors in most current AI implementations, these “individual behaviors” are also the most highly developed.

“Team-enabling” behaviors are the most critical behaviors from the teamwork perspective. In these behaviors, we include Communication, Synchronizing Actions, and Team Member Monitoring. Team Member Monitoring is similar to Self Monitoring in that it is the process of evaluating how close another agent in the team is to completing a task. This behavior, in conjunction with the other individual monitoring behaviors, enables the Synchronizing Actions behavior, in which an agent modifies its individual behaviors as needed in order to match its actions to those of its fellow team members. As a simple example, in a unit moving in formation, one agent might adjust its speed after observing the relative location of its neighbor. From the individual agent stand point, it is important to note that these “team-enabling” behaviors present an agent with a set of courses of action that may be suboptimal from the perspective of their own personal welfare. For a fighting agent whose goal it is to kill an enemy, the behavior of Communication may cause him to engage in actions other than those which strictly benefit him in the direct accomplishment of his goal. To communicate a message, the agent may have to move to a location which is not the optimal tactical location in a given situation. At the very least, it makes the agent divert cognitive resources to an act which is not directly involved in accomplishing a goal (assuming the act of communicating itself is not the goal). All of the “team-enabling” behaviors have that trait of possibly being sub-optimal choices in any given situation for an individual agent. These behaviors, by their nature, sacrifice the near term good of the individual for the long term betterment of the team. This is a fundamental aspect of teamwork.

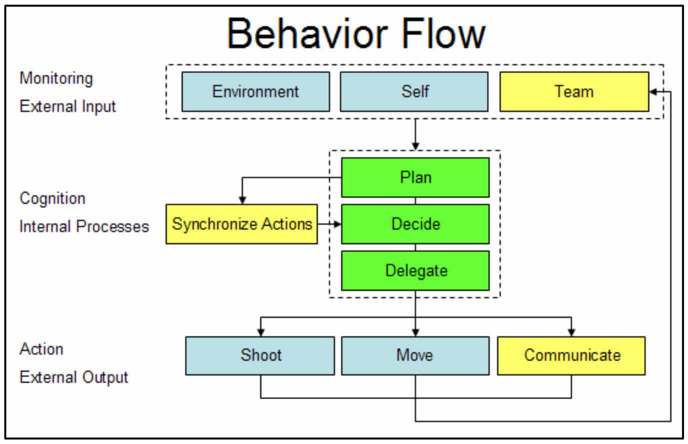


FIGURE 3. BEHAVIOR FLOW

To understand how these behaviors interact within an individual agent, we have arranged the agent behaviors into a “behavior flow” (see Figure 3), structured on the “input, process, and output” (IPO) framework (Kendall and Salas, Measuring Team Performance, 2004). The “input, process, output” model is a reductionist approach for examining a system. We will use this framework as a convenient analysis method that allows us to categorize the behaviors based on how they interact within the agent ‘system’. While this categorization of behaviors may be considered orthogonal to our previous categorization of “leadership”, “individual”, and “team-enabling”, it provides us a sequence for how the behaviors may be engaged in a single “thought cycle”. This is a part of the foundation for an in-depth discussion of the model representation to a level of detail or understanding that would facilitate the writing of a computer program implementation.

In our model, the inputs and outputs both consist of a combination of “individual” and “team-enabling” behaviors. Inputs are the Environment and Self Monitoring “individual” behaviors, and the Team Member Monitoring of “team-enabling” behaviors. Outputs include the “individual” Shooting and Moving behaviors, and the Communicate “team-enabling” behavior. The internal processes contain the entirety of the “leadership” behaviors (Planning, Deciding, and Delegating) along with the Synchronize Actions “team-enabling” behavior. While the “leadership” and “individual” behaviors squarely divide themselves between internal and external processes, the “team-enabling” behaviors can be found in each step of the IPO framework.

As an example of how the behavior flow might work, consider the following scenario (see Figure 4) in which agents from a team are moving upon an objective. Each agent is delegated the task, by their mutual leader (who could be any of the agents depicted) to move from the box on the left, marked S, to one of the boxes on the right, marked E. Each agent is given the task to move along a separate route as quickly as possible.

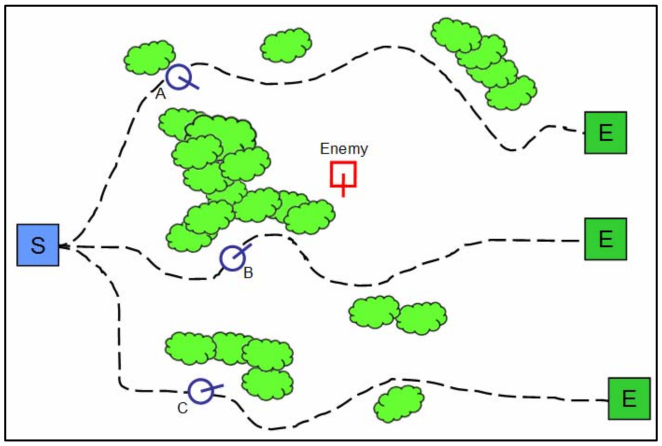


FIGURE 4. TEAMWORK SCENARIO

At some point during their movement, Agent A observes an enemy (depicted by the red square) through his Environment Monitoring behavior. Due to the intervening obstacles, Agents B and C have not seen the enemy agent yet. Agent A assesses that the enemy is not oriented towards him, and will not see him if he continues on his assigned route. Through his Team Monitoring behavior, he also assesses that the enemy does pose a threat to Agents B and C. In the strict optimization of accomplishing his assigned task quickly, the rational choice might be to ignore the enemy. The enemy does not impede his ability to reach his green box, and perhaps attacking the enemy will require slowing his own progress towards his goal. However, based on the "team-enabling" behavior sets, he is now presented with some additional behaviors, specifically, Team Monitoring (already enacted), Synchronizing Actions, and Communicating. It is worth noting that none of these actions will help Agent A reach his green box, and in fact, may even slow his progress. However, to demonstrate teamwork, Agent A needs to either alert his other team members to the presence of the enemy or remove the threat himself (or do both). Whichever of his specific actions he chooses to take will be dependent upon other aspects specific to the implementation. But the important concept is that the Agent is taking an action which is not optimal for his own goal accomplishment, so that his team might better be able to meet its team goals. It is the subordination of the individual for the good of the team, and it would not be possible without the "team-enabling" behaviors.

In summary, this section has presented a structure of behaviors which comprise teamwork, as well as a model demonstrating how those behaviors can work together within an individual agent. It is our hope that this model will be useful in providing structure to the concept of teamwork as well as some viable methods for implementing teamwork behavior in agents. Additionally, we hope that, by modeling teamwork within our framework, one will be better equipped to explore the phenomenon of teamwork and those interactions which lead to resultant team successes.

IV. SOFTWARE DESIGN AND IMPLEMENTATION

A. INTRODUCTION

In this chapter, we will discuss our implementation of our abstract teamwork model. We will also discuss in some detail the design considerations, the coding process, and the implementation objectives.

Our implementation goal was to develop an agent-based simulation which allows us to create agents which utilize our model in order to create inter-agent teamwork dynamics. We initially considered the possibilities of implementing our model within an existing simulation, such as the U.S. Army's Objective One Semi-Automated Force (OneSAF or OOS) or even Epic's Unreal Tournament Series of games. However we opted for creating our own model to avoid confounding factors which might be present from existing behavior frameworks and agent functions. By building an application for us to manipulate, we trade off the additional overhead of creating an entire simulation with the assurance that only our work would go into the creation of the agents and their behaviors. The end result is a very simple simulation which lacks some of the functionality of a more mature program, but of which we can be sure the results are entirely due to our own inputs.

B. DESIGN CONSIDERATIONS

1. Programming Language and IDE Selection

After some deliberation, we chose to create our application using Java 2. We made this decision understanding that this would be a very simplistic application that would depend upon code optimization or efficiency. Additionally, the "error checking" and automatic garbage collection in Java allowed us to focus more on the content, rather than the syntax, of our code. Sun's extensive website and the Application Programming Interface (API) for Java 2 also made it easy for us to reference new functions and classes as the need arose. All of our code was written within the NetBeans version 4.1 Integrated Development Environment (IDE) running the Java Runtime Environment (JRE) jre1.5.0_06. Originally, this download was offered as a package included in the Java 2 Java Developers Kit (JDK) jdk1.5.0_05 coupled with the NetBeans 4.1 download from Sun's website. NetBeans has been updated to version 5.0; however, for continuity sake and to avoid any possible backwards compatibility problems, we finished our development within NetBeans version 4.1.

Some other factors which drove our decision to utilize Java 2 and the NetBeans IDE include, but are not limited to the following: easy to use Input/Output packages, versioning support within NetBeans, additional web service support for Java 2 and the ability to utilize other Java packages such as the Java Architecture for XML Binding (JAXB). We especially found the integration between NetBeans and the TortoiseCVS application useful in facilitating multiple programmers working at once.

2. Versioning

In order to effectively distribute the programming workload, we found it necessary for both thesis members to be able to edit the code at the same time. For this reason, we decided to utilize Concurrent Versioning System (CVS), specifically TortoiseCVS. With this tool we were able to post our developmental code on a server maintained by a professor at the Naval Postgraduate School. This enhanced our team's ability to track and coordinate source code changes throughout the development process. Additionally, we were able to add branches and tags allowing us to save all the code through a specific developmental stage or milestone. This provided the capability to revert back to the latest, fully-operational version on the server. As noted earlier, NetBeans, like many IDE's, fully supports CVS functionality directly within the editing environment.

C. SIMULATION FRAMEWORK

1. Basic Operation of Simulation

The application is built around an event timer which runs a thread every specified number of milliseconds (currently 100). This thread executes a "turn" method of a subclass called SimEngine. The SimEngine, in turn, contains the hash maps which hold the agents within the simulation. Each turn provides the inputs and outputs for each agent, updating their internal variables and calling their member functions as appropriate. The abstract notion is that the turns process in three distinct phases: 1) the environmental actions, such as shots, vision, and auditory messages, then 2) the cognitive internal processes, and lastly 3) the agent actions. In detail the turn method calls other member methods of the SimEngine in the following sequence:

- receive input from the application user (i.e. add in new agents)
- check line of sight for each agent and process "see" perceptions as appropriate
- process the SimEngine internal "shots fired" data structure
- process the perception of "being shot" for agents
- as appropriate, process messages being verbally exchanges between agents
- execute the agents' internal cognitive processes
- add in new shots based on agent actions resulting from their cognitive process
- move agents, again, according the actions resulting from their cognition

The multithreaded nature of our program makes it inherently important that each turn is completely executed before the next turn thread is begun. The use of hash maps greatly enhanced our efficiency, and we have had little problem staying within that constraint. Most turn executions take on the order of 16 to 30 milliseconds. This is a great improvement over the results we achieved when just using ArrayLists, where concurrent threads frequently interfered with simulation functions. Even with hash maps, however, it would not be too difficult to overwhelm our simulation by adding many agents. Due to the initial and exploratory nature of our work, though, we have mostly restricted our teams to groups numbering between four and five agents. Undoubtedly, there are more effective methods of structuring our application, but at this time, this method adequately serves our needs.

2. Design of Graphical User Interface (GUI)

The event timer which controls the SimEngine is a part of a standard JFrame, which in turn manipulates a JPanel. The JPanel is the where the user input is taken in from the mouse. It is within the JPanel that the user can add and remove agents, as well as add walls and issue a simple order to give direction to the agents. All of these inputs are provided through a mouse Listener, which temporarily stores the data until it is passed into the next SimEngine turn. As noted above, the SimEngine's first action during a turn is to update its internal data structures from the input provided by the JPanel.

The user is provided with a set of options at the top of the panel. The primary choice for the user is to select between red and blue agents. Once the selection is made, clicking in the main panel will create a new agent. Clicking on an existing agent of the right type (red or blue) will remove that agent, and clicking and dragging will move an agent. Another frequently used GUI feature which directly relates to the agents is the Start/Stop button. In essence, this button controls a Boolean value which determines whether or not the SimEngine 'turn' methods are called. If the 'turn' methods are not called, then a substitute 'pauseTurn' method is called in its place. The 'pauseTurn' still allows components of the SimEngine to be updated, such as the addition of new agents to their appropriate hash maps, but does not process the agent behaviors, or any of the other environmental features, such as shots or messages.

The user can also create walls, by selecting the choice from the panel, and then clicking twice to set the start and end point of each wall. Walls block agent's movement, vision, and gun shots. However, we discovered that while conceptually relevant, the walls also add a requirement for a more robust navigation system for the agents which we have not yet implemented. Therefore, though they are functional, the walls are not used in our experimentation to test our agent behaviors.

Another useful feature of our GUI is the ability to store and load scenarios. As mentioned above, we utilize the JAXB API to extract specific information from our SimEngine and store the data in XML files. We can then use these files to replay specific scenarios repeatedly. The outcomes of simulation are variable. Therefore, it is sometimes beneficial to repeatedly observe the outcomes of the same scenarios. We also utilized this feature in trouble shooting our code. At this point, the simulation does not have the ability to lock the random seed to replicate outcomes.

Last, but not least, is the ability to add a ‘movement order’ to the SimEngine. The ‘movement order’ is a means of providing guidance to the agents beyond their initial positioning without directly manipulating their location. The order consists of a direction and a location, and is input with two clicks of the mouse, similar to how a wall is made. These two pieces of information, direction and location, are then passed onto the lowest id-numbered blue agent (who should be the team leader). This is somewhat similar to how many ‘real-time strategy games’ issue user orders to the player’s units. The end result of the order is that it becomes a location where the agent ‘team’ should move to, and a direction in which they should orient once they arrive at that location. Though we did not make extensive use of this feature, it does provide a good starting point for hierarchical aggregation of units. Once a team leader can accept a simple two variable ‘order’ from the user, it will also be able to accept a similar ‘order’ from another agent.

D. AGENT DESIGN

1. Agent Architecture

One underlying motivation for our agent structure is our desire to move beyond a purely reactive agent. It remains inescapable that some of our agent’s behaviors are suspiciously reactive in nature. For example, an agent who observes an enemy will “react” to that perception. In the construction of our agents, however, we have sought to incorporate some layers of control which in effect decouple the stimulus and response. In decoupling the “stimulus-response” behaviors with intervening cognitive layers, we create a more realistic model of human cognition and behavior.

Russell and Norvig go on to classify architectures that combine reactive and deliberate techniques as hybrid architectures. They also specify a type of hybrid which they call the three-layer architecture consisting of reactive, executive and deliberate layers (Russell and Norvig, 2003, pg 933). Our agent implementation contains components which roughly correspond to each of these layers.

The reactive layer handles very low level behaviors which most likely are temporary in nature. The prime example of this is the fratricide prevention check. This check is executed by a method that evaluates the position of teammates and overrides the agent's shooting action if there is a friendly agent who is within a "danger zone" around the shooting agent.

The executive layer then is the middle management, which governs the continuous behavioral choices of the agent. The best instance of this is in the agent's decision to move, shoot, and/or communicate.

The deliberate layer plays out in the team leaders, who based on their world model, will send orders to change the formations of the team to a tactically appropriate formation. In turn, it is also evident in the team members, who based on those orders from their leader, determine where they should move to, and take the appropriate actions to get themselves into positions.

In addition to these architectural layers, we also sought to add another conceptual layer to decouple our agents' perception and actions by giving our agent competing motivators. While the use of the contact list combines the agent's perception of the world in its *current* form with its incoming perceptions, the use of motivators provides a mechanism to incorporate an abstraction of the agent's world model *over time*. The motivators serve as semi-persistent abstractions of the agent's world model, which impact its decision making cycle.

2. Motivator Based Structure

We began by extolling the virtue of assuming away the emotional side of leadership. Now, in a limited sense, we are attempting to give some emotional capacity back to our agents. Specifically, we have given our agents three "emotions" or motivators to serve as their goals. These motivators are Aggression, Discipline, and

Insecurity. During each cognitive cycle (every 100ms), the agents perform various mathematical functions on the data contained in their ‘contact list’ world model. These formulas condense the array list of information into three double precision numerical variables, ranging between 0 and 100, which reflects the strength of the motivation at a particular point in time. Additionally, we implement a ‘Low Pass’ filter for each motivator, in essence creating an internally contained ‘history’ of the agent’s feelings. This serves as a buffer to prevent an agent’s motivators from solely reflecting the current world state, and giving some persistence to the ‘emotional’ aspect of the agents. From this perspective, emotions can be viewed as a cumulative representation of the agent’s environment. An agent that has just had half of his teammates killed will view contact with new enemy much differently than an agent who has not. Even if the force ratios are similar in each situation, the events which lead up to the encounter shape the agent’s “emotions” or motivations, which consequently affect it’s behavior.

During each cognitive cycle, the agent’s apply the following formula to condense the data contained in their contact list. In defining what contributes to each motivator, we quickly developed a broad list of factors which might contribute to each “emotion”. For simplicity sake, we then paired down the list by limiting each emotion to its first order impact, and by simultaneously trying to eliminate redundant factors between motivators. For example, an injury to an agent might intuitively appear to increase insecurity as well as decrease discipline. However, we considered the discipline to be a second order effect. The injury makes an agent more insecure, and therefore, insecurity is raised, making discipline lower valued in comparison. Therefore, the first order effect is to insecurity, and by default, the second order effect is to discipline. Obviously, there are many factors that could be considered for each motivator, but for simplicity sake, we sought to limit ourselves to between two and four factors each. As a disclaimer, there are many different ways in which someone might quantify the emotional effects of an environment upon an agent. We in no way are supposing that we have the best solution. Instead, our goal here is to provide a dynamic and somewhat sensible goal-oriented aspect to the agents as a backdrop against which to test our teamwork theory.

a. Aggression

Aggression reflects the agent's desire to engage the enemy and is dependent on two elements. The first is a weighted proportion of the injured and uninjured friendly agents, where a higher the ratio of friendly casualties to enemy casualties increases aggression. The second element is inversely proportional to the distance of the agent's target (the closest enemy) with respect to the agents desired optimal distance from that enemy. The closer the agent is to his enemy, the greater his aggression. As their sum totals a value on a scale of 0 to 100, we arbitrarily weighted each component equally responsible for potential 50 points of aggression. The general mathematical formula is:

$$50 \times \frac{(\#\text{Dead} \times 2) + (\#\text{Injured})}{\#\text{Dead} \times 2 + \#\text{Injured} + \#\text{Alive} + 1_{\text{self}}} + \left[50 - \left(50 \times \frac{\text{OptimalDistance}}{\text{DistancetoTarget} + \text{OptimalDistance}} \right) \right]$$

Below is a portion of code which recalculates the aggression variable for use by the agents:

```
//aggression, same incremental model
double newAggression = 0;
//ratio of uninjured to injured/dead friendly
newAggression+=50*((friendlyDead*2)+friendlyInjured)/
                  ((friendlyDead*2)
                  +friendlyInjured+liveFriend+1));
if (target!= null)
newAggression+= 50 - (50 * (AgentRef.OPTIMAL_DISTANCE /
                  (distanceTo(target)+
                  AgentRef.OPTIMAL_DISTANCE)));
aggression = (aggression *oldRatio) + (newAggression * newRatio);
```

b. Insecurity

Insecurity reflects the agent's desire to not be killed and has four factors. The first is whether an agent is injured, and an injury increases insecurity. We considered there to be no difference in this case between mobility and firepower injuries.

The next factor is how many shots the agent detects within his immediate vicinity. We chose to use twenty-five as a baseline for the maximum expected number of shots that any one agent might encounter in a snapshot in time. The more shots in the vicinity, the more insecure an agent is. We also considered the ratio of live friendly agents to live enemy agents and the distance the agent is to its team mates, abstracted as their center of mass (COM). The farther an agent is from his team, the more insecure he will feel. Once again, we weighted each factor equally for 25 insecurity points.

$$(25\times \mathrm{injured})+\left(25\times\frac{25}{\mathrm{shotsInVicinity}+1}\right)+\left(25\times\frac{\mathrm{liveEnemy}}{\mathrm{liveEnemy}+\mathrm{liveFriendly}+1}\right)+\left(25\times\frac{25}{\mathrm{distanceTo}(\mathrm{friendlyCOM})}\right)$$

Below is a portion of code which recalculates the insecurity variable for use by the agents:

```
//insecurity, incremental - half old, half new
double newInsecurity = 0;
//health
if (health != AgentRef.LIVE)
newInsecurity += 25;
//shots in vicinity
newInsecurity += (25 - (25/(shotsInVicinity + 1)));
//number of live enemy to number of live friendly
newInsecurity += (25 * (liveEnemy/(liveFriend + liveEnemy + 1)));
//distance to friendly COM
newInsecurity += (25 - ((25/(distanceTo(friendCOM)+1))));
insecurity = (insecurity * oldRatio) + (newInsecurity *
            newRatio);
```

c. Discipline

Discipline reflects the agent’s desire to follow orders. We narrowed discipline to three factors; the distance to the leader, the health of the leader, and what we termed the ‘commo lapse’. The distance to the leader is measured in terms of the communications range, the maximum distance that an agent can hear a message. The closer an agent is to his leader, the more disciplined he is. The health of the leader is a simple binary value depending on whether the leader is live or injured. If a leader is injured, we conjectured that the agent will lose faith in the leader, and therefore be less disciplined. The ‘commo lapse’ is the time since the agent has last heard from any of the other agents. If an agent has not heard from any of his team mates in a while, his discipline will decrease. As per the above, the factors are weighted roughly equally, with the distance to the leader gaining an extra point for a total of 100. It should be noted that the leader gets 67 discipline points for the first two factors.

$$\left(34-\left(34\times\frac{\text{distanceToLeader}}{\text{distanceToLeader}+\text{commoRange}}\right)\right)+$$

$$\left(33\left(\text{ifLeaderLive}\right)||16\left(\text{ifLeaderInjured}\right)\right)+$$

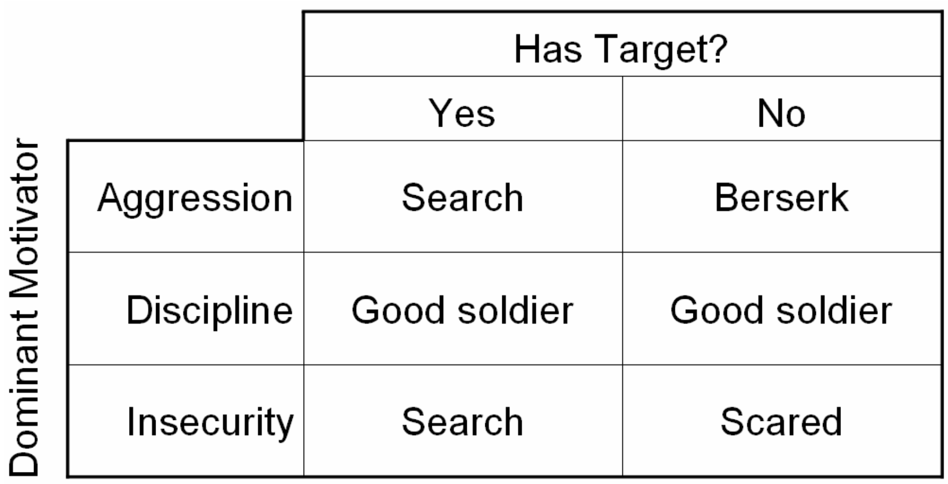
$$\left(33-\left(33\times\frac{\text{commoLapse}}{\text{commoLapse}+\left(\text{commoTime}\times 2\right)}\right)\right)$$

Below is a portion of code which recalculates the discipline variable for use by the agents:

```
//discipline, same incremental model
double newDiscipline = 0;
if (teamLeader==idTag)
//if the agent is the leader, give full credit for health
//and distance
newDiscipline += 67;
else  {
       //distance to leader
```

```
        doubledistanceToLeader=distanceTo(ellipseToPoint(((Contact)
                          contacts.get(teamLeader)).location));
        newDiscipline+=(34-(34*(distanceToLeader/
                      (distanceToLeader+AgentRef.COMMO_RANGE))));
        //leader health
        if (((Contact)contacts.get(teamLeader)).strength ==
                      AgentRef.LIVE)
              newDiscipline += 33;
        else if (((Contact)contacts.get(teamLeader)).strength! ==
                      AgentRef.DEAD)
              newDiscipline += 16;
        }
  //time from last commo, max out at 10,000 ms (10 seconds)
  long commoLapse = Calendar.getInstance().getTimeInMillis() –
        timeOfLastCommo;
  newDiscipline += (33 - (33*(commoLapse/(commoLapse +
        (AgentRef.COMMO_TIME*2)))));
  discipline = (discipline * oldRatio) + (newDiscipline *
        newRatio);
```

During each cognitive cycle, the agent assesses his motivators and updates their values. During the later action portion of the agent's think "method", the dominate motivator is used to select what behaviors the agent will enact. We have created three abstract modes of operation, or complex behaviors, to capture the difference between each dominant motivator. In order to prevent confusion between the "abstract behaviors" and the more atomic behaviors we defined in our model (Move, Shoot, Plan, Communicate, etc.) we will refer to these three abstract behaviors as modes. An agent will choose a mode based on its dominant motivator and the presence or lack thereof of any enemy. The dominant motivator is the one whose value has been highest recently. The table below illustrates how the modes are selected.



Dominant Motivator		Has Target?	
		Yes	No
	Aggression	Search	Berserk
	Discipline	Good soldier	Good soldier
	Insecurity	Search	Scared

TABLE 1. AGENT BEHAVIOR MODES

In 'Search' mode, an agent will remain stationary and turn left and right, depending on where he perceives his team to be, in an effort to gain more information about his environment. In 'Good Soldier' mode, the agent will move to his designated position in a formation, based on his perception of who his leader is, and will shoot at enemy when they enter are in his field of vision. A 'Good Soldier' prioritizes moving into formation over turning to shoot the enemy. In 'Berserk' mode, the agent simply moves directly towards the nearest enemy, regardless of orders, formation, or friendly location, while shooting at the enemy. A 'Scared' agent moves away from his perceived enemy center of mass, also regardless of orders or formations.

3. Demonstrating Teamwork and Team-Enabling Behaviors

Having designed our agents with a hybrid-architecture and a motivation-based structure that drives their behavior, we come back to the core of our theoretical teamwork model, which focuses on the "Team Enabling" behaviors as we have defined them. To facilitate our own experimentation with these behaviors, we incorporate percentage values into the agents to define each agent's propensity to engage in the behaviors of interest. Accordingly, each agent has a value for Communicate, Synchronize Actions, and Team Member Monitoring. During the cognitive processes, the agent will conduct a random number check against these percentages, and if the check passes, then they will proceed with the behavior, otherwise, they will bypass it.

Communication is straightforward; the agent will either send a message, or it will discard it. Synchronizing Action determines whether or not an agent will account for friendly agents when determining it's facing and if it will update its relative location in formation with its team. Team Member Monitoring determines if an agent will process message regarding friendly positions, and if it will update friendly status based on the message traffic that it hears. In order to always carry out these behaviors, the values for each percentage are set to 1.0, and to never execute them requires a value of 0.0. Any value in between will give a random sample in which the agent's will sometimes execute the behaviors and sometimes not.

4. Tactical Infantry Cognitive Artifacts

In order to understand the behavior of the agents, it is necessary to discuss some of the details about algorithms and cognitive artifacts that have been included in their structure that relate directly to their role as infantry style combatants on a battlefield. In each case, we have sought to include enough depth to allow for realistic and hopefully interesting behavior on the part of the agents, but at the same time balance that with the desire to keep things simple. Any of these aspects of the agents could be expanded upon greatly, but we consciously limited complexity so as to focus on our team model, as opposed to focusing on the mechanics of our simulation.

a. Formations

Formations are a fundamental building block for military units. Common knowledge of formations and roles/positions within a given formation is expected of all soldiers. Each agent has algorithms to define four formations. The formations might be described as procedural; each depends on the number of agents in the team and the individual agent's rank within that team. Thus, the same formation algorithms can define locations and orientations for teams of any number. It is important to note that in keeping with our agent oriented teamwork, the formations are predicated upon the agent's individual awareness of his teammates and their locations, which may be inaccurate. Thus, when a formation is achieved correctly, it is because each individual agent shares similar understanding of their environment. For simplicity sake, agents will default to the coil formation.

The four formation algorithms each agent possesses formations are the Coil, Line, Staggered Column, and Wedge. The Coil and Line are meant to be defensive formation used when the team is stationary. The Coil is the default formation, and is taken in the absence of enemy. The Line formation is taken when enemy is identified, and the team can best leverage it's firepower by coming abreast of each other on line and avoiding each other's line of fire. The staggered column is an offensive formation, taken when the team is moving, and there is no enemy. Lastly, the wedge is also an offensive formation, but is taken when the team is moving towards enemy. Similar to the Line, it reduces the chance that agents will get in each others line of fire. The formations are depicted below, with the team orientation for each shown formation being towards the bottom of the page.

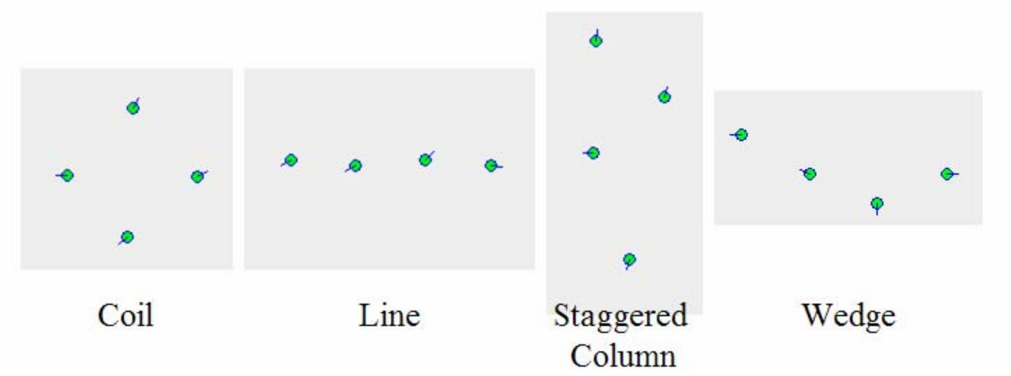


FIGURE 5. AGENT FORMATIONS

b. Leader Selection

While the military does impose a form of external "hive mind" mentality by designating leaders through systems of rank and orders, our agents needed a simple way in which they could dynamically recognize who is their leader without being told so by an external entity. In order to do so, we impose a simple system where a perception of an agent also includes the agent's unique ID number. If the agent recognizes that another agent has a lower number, it will identify that agent as its leader. It is possible that at a given time, due to imperfect understanding of the environment, agents may recognize different leaders. However, once all information has been disseminated and all agents are aware of each other, each side (red or blue) will ultimately have one leader. The knowledge of ID numbers also comes into play when calculating the rank of an agent relative to his peers. This helps him determine his location within the formations described above.

c. Messages

Capitalizing on the abilities of the Java2D API, our simulation uses Circle2Ds to represent communication. It takes an agent a set amount of time to "send" a message, and for simplicity sake, all of our messages take 2.5 seconds to pass. When the message is initiated, the SimEngine checks which agents are located within the Circle2D which represents the distance over which the message can be heard. Throughout the transmission of the message, the Circle2D is repeatedly checked, and if any of the original agents are no longer within the Circle2D, they are removed from the recipient list. Once the 2.5 seconds are done, those agents remaining in the recipients list will "receive" the message.

d. Contacts

A contact is a class defined to contain information about an agent. Its class structure makes it easy to encapsulate perceived information, add it to an agent's situational awareness, and pass it to other agents through messages. A typical contact will have information about an agent's ID number, location, and health. An agent's situational awareness is composed strictly of a list of contacts. These contacts are obtained either through direct observation, or indirectly by hearing the contact in an “auditory” message.

e. Communications Protocol

The agents all obey a simple protocol for determining when to send a message. Initially, they will send a contact report the first time they perceive another agent, enemy or friendly. After the agent has been perceived and added to the situational awareness, they will send a new message only if the state of that agent changes (such as its health). They also will only send a message if there is a friendly agent within communication range to receive the message (though no guarantee is made that the agent will remain within communication range for the duration of the transmission). The only other event which spurs an agent to communicate is if the agent is a leader, and he decides there is a need to change formations.

When an agent hears a message that is a contact report (i.e., another agent reporting information about a third party), the agent effectively gains two contact reports. One is about the subject of the contact report (the third party), and the other is about the sender of the message. This second contact about the sender is more limited, and only includes the ID number and the location of the sender, but it does update the agent’s awareness of his environment. If both new pieces of information meet the right criteria, then they may in turn each result in a new message being sent out by the receiver. The effect is that until a group of agents all has similar knowledge and no agent is hearing anything unknown to them, they will queue and emit a stream of messages. Typically at the beginning of a simulation when agents are forming their team hierarchy and seeing who is around them there will be a period of intense “chatter” as each agent perceives new information and repeats it. Once the information is synchronized, then no agent will perceive anything new and the “chatter” will die out.

f. *Search Patterns*

To capitalize on the benefits of having teammates nearby, when the agents enter into a search mode they will reduce their field of responsibility proportionally to the number of agents present. For example, an agent by itself will search all 360 degrees around himself, which two agents together will each only scan 180 degrees.

g. *Fratricide Checking*

When an agent has decided to shoot an enemy, one of the last things his cognitive methods will do is check his situational awareness (his contact list) to make sure that there is no friendly agent within a certain number of degrees to the left and right of his target or within a small circle around himself. This method has proven almost completely successful in preventing fratricide. There is still a possibility, however, if an agent's awareness of the environment is incomplete, that he will still engage an enemy even though a friendly agent may be in his line of fire.

E. DESIGN FOR ANALYSIS

The design of the input and output for our simulation proved to be a very important decision point in the coding process. We wanted the capability to save and load scenario files. This would allow us to conduct as many runs of the simulation as desired, each with its own output file which could then be used for analysis. We determined that the easiest way to accomplish this was to use Extensible Mark-up Language (XML) scenario files. This would allow any user, even with limited knowledge of our simulation, to generate scenario files which could easily be read by our code.

This was accomplished by drafting an XML schema for our program. The schema lays out exactly what elements are required in a scenario file which could then be populated by corresponding attributes, as desired, by the user. Once this schema was generated, any user could take our schema and generate valid, well-formed XML scenario files that would populate our simulation for scenario runs. Likewise, our code would generate valid, well-formed XML scenario files representing any stage of the simulation, for example the end-state after an entire run. These files could then be sent back to the user for further analysis.

We were able to add the above described functionality by incorporating the Java Architecture for XML Binding (JAXB). This package is included in the Java Web Services Developer Pack and can be downloaded from Sun’s Java 2 downloads website. We utilized version 1.5. The JAXB package is included as a jar file, which we added to our project in NetBeans by adding it as a library.

V. QUALITATIVE ANALYSIS

A. INTRODUCTION TO ANALYSIS

This chapter will present the analysis of our simulation, but will not attempt any comparisons of our simulation with any existing Department of Defense (DoD) Simulation. Our goal is to merely verify the behavior of our agents with respect to our abstract model from Chapter III. Additionally, we will conduct some qualitative analysis of our agent behavior while varying the team-enabling behaviors within the simulation. This will serves as a method to verify the impact of our model of teamwork.

This analysis will consist of two parts: first, an analysis of the teamwork behavior motivators, and second, a qualitative analysis of the agent behavior when varying the existence of the different team-enabling behaviors within each agent. This analysis will provide the reader a more in-depth understanding of how to implement our abstract model. While the focus of this research was to create an implementation of our abstract teamwork model to aide in the presentation of our proof-of-concept, we hope that this brief analysis will benefit those who may wish to make improvements to the existing simulation or possibly even, serve as a catalyst for future, more robust implementations of our model.

B. ANALYSIS OF TEAM MEMBER MOTIVATORS

The purpose of this analysis is to qualitatively explore the agent motivators. The goal is to verify the general deterministic nature of the motivators and to identify and document any unexpected or unusual behaviors and their suspected causes to aid in improving future versions of the simulation. While the motivators are intended to be dynamic throughout, it is expected that some improvements can be made to the motivator equations to allow agents to better transition between behavior modes as expected. Again, this first analysis does not attempt to validate this simulation nor does it serve as an attempt to prove that this implementation is the best representation of our abstract model. We show that our agents behave as expected, or at least as designed, when directed by a leader both in the presence and in the absence of enemy agents.

This analysis consists of a qualitative exploration of a series of scenarios. The scenarios are not intended to mirror actual military situations; however, they were designed to represent certain situations which serve to isolate the different motivators, with the intent of exploring the boundaries of each. We conducted the analysis in three parts: first, by developing and describing several scenarios which attempt to fix the motivators and reporting on the results of successive runs of these scenarios, second, by conducting a brief mathematical and graphical analysis of the motivators using Microsoft Excel with the intent of improving each motivator equation, and third, by describing the original scenarios and reporting on the effects that the improvements to the motivator equations have on the overall agent behaviors. The general design focus of the scenarios was to create situations where obvious behaviors should occur in order to record successive runs and assess the adequacy of the simulation's fidelity. For our purpose, the analysis of the motivators serves as our verification of our simulation and model. At a minimum it is an excellent source of documentation for future verification efforts.

1. Scenario Development and Fixing Motivators

In order to adjust the motivators, specific scenarios were designed to analyze each one. The three xml scenario files used in this part of the analysis are called: highEnemyRatio, distToLdr, and distToCOM and can be found in the distribution directory within our source code. The start set for each as well as the detailed description of agent behavior during successive runs follows.

a. Aggression

A logical choice of scenario design to observe the aggression motivator was to place a small team in the presence of a large enemy force. We called this scenario, highEnemyRatio, Figure 6. In this scenario, one can see a large enemy force centrally located on the panel with a split friendly force. For this scenario, the leader is isolated on the left and the other two agents on the right are team-members. This will allow us to see the dynamics of the behavior motivators in addition to isolating aggression. At simulation start, we would expect to see the agents start in Berserk mode and then transition to either Good Soldier or Scared Mode, depending upon the lapse in reporting of the presence of enemy agents. Eventually, when all enemy are reported, we would expect the agents to transition to Scared mode due to the overwhelming size of the enemy. In Scared mode, they should immediately start moving away from the enemy agents.

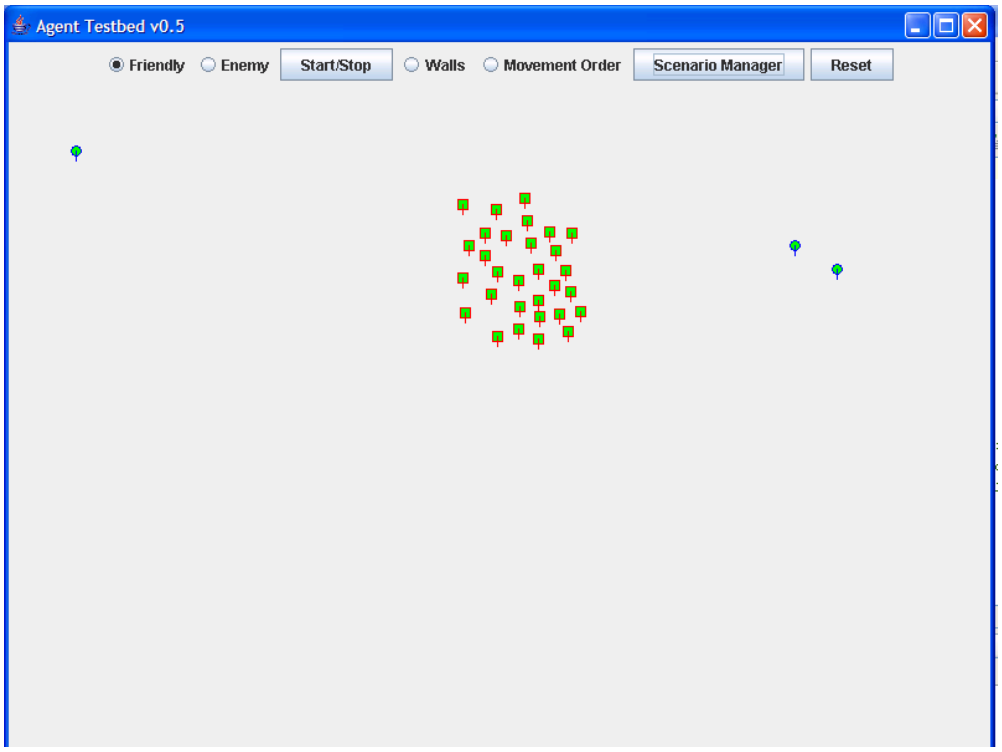


FIGURE 6. SCENARIO FILE: highEnemyRatio.xml

Initially, with all motivators turned on, there was no way to get the agents to move into Berserk mode. This was because the discipline equation was not dynamic enough. Initially, in almost every scenario, discipline dominated for the entire run and the agents remained in Good Soldier mode. We will discuss this later when we discuss the discipline equation analysis. Thus, in order to isolate the aggression motivator, we set the discipline value equal to zero and ran the simulation with the enemy behaviors turned off. With discipline set to zero, the friendly agents start in Berserk Mode. In fact, the aggression value remains the highest value until enough enemy are reported to drive the insecurity value higher. We will analyze the aggression value further in the mathematical analysis below, but for now it is clear that it is not dynamic enough to sufficiently drive the aggression behavior of the agents in a believable manner.

b. *Insecurity*

Here, we looked at two scenarios, Figures 6 and 7, to initially sort through some of the potential issues with our insecurity formula. The start set for each as well as the detailed description of several runs follows.

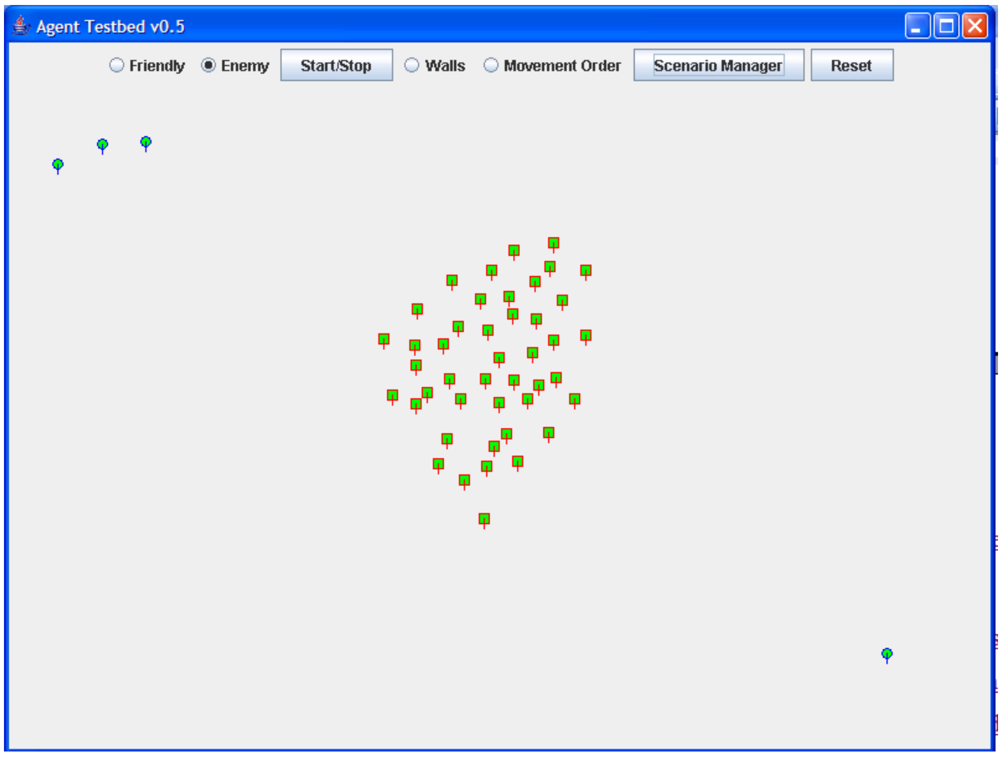


FIGURE 7. SCENARIO FILE: distToCOM.xml

In Figure 7, we see another scenario with the team split by a large enemy force. In this case, the intent was to isolate the communications aspect of the insecurity motivator. The leader is co-located with two team-members in the upper left corner and a lone team-member is isolated from the rest of the team. We describe runs below both with and without enemy agents present in this scenario. The screenshot with no enemy present is not presented. With no enemy present, the lone agent should relocate to join the other team members and all agents should remain in Good Soldier mode throughout the run. With enemy present, however, the lone agent should feel cut-off from the team and transition from a brief stint in Berserk mode to Scared mode as the large enemy force is reported or discovered.

Initially, with no enemy present, the lone friendly agent's insecurity value is smaller than those of the three agents that are close to one another. This is due to the communications lapse in reporting all agents and their locations. Once the lone agent discovers the presence of the other agents and their respective locations, its insecurity value quickly elevates and is the highest of the three. When enemy are present, the results are similar. In this case, when the lone agent becomes aware of the existence of the other friendly agents and the all enemy agents are reported, its insecurity value also goes up accordingly. This effectively demonstrates that the lone agent feels isolated and is cut-off from the other friendly agents by the enemy. The resultant behaviors are as designed and expected. It is interesting to note that the increase in the insecurity values after the enemy is reported are negligibly higher and possibly not large enough to make a difference in the overall behavior of the agents. This will be addressed in further detail in the mathematical analysis of the equations.

For the remainder of our discussion of insecurity, we again return to the scenario in Figure 6, highEnemyRatio. As with the aggression analysis, in order to effectively isolate insecurity, we set the discipline value equal to zero and then ran the simulation with the enemy behaviors turned off. With discipline set to zero, the friendly agents initially start in Berserk Mode. As described above, the aggression value remains the highest value until enough enemy are reported to drive the insecurity value higher. As expected, due to the presence of such an overwhelming enemy force, the friendly agents transition to the Scared mode and consequently move away from the large enemy force. This behavior is inconsistent with the size of the enemy force present. We would expect that the two agents on the right should enter scared mode immediately, since they are isolated from the leader by the enemy force.

When this scenario is run with the enemy behaviors turned off and the friendly agent's discipline values enabled, the resultant friendly agent behaviors are also more in line with what is expected. Once contact is made with all friendly agents and the large enemy force is identified, all agents' motivator values are relatively high, compared to runs of scenarios with much less enemy and with the team members located in close proximity to one another. However, there is such a disparity between the discipline values and the values of the other motivators that the agents never transition to any mode other than Good Soldier. Because of this, in this scenario, the two agents on the right very often move directly through the large enemy force, sometimes without firing a single shot. When the enemy behaviors are turned on, as one can imagine, the results are quite catastrophic. Below is a snapshot from the output of the agent motivator values in this scenario:

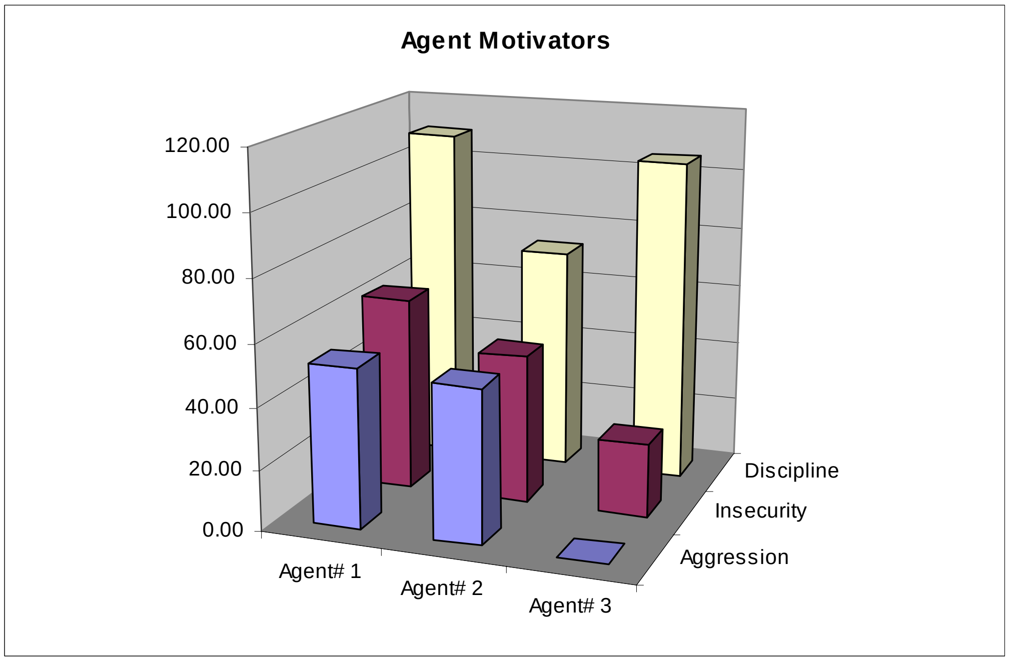


FIGURE 8. SNAPSHOT OF AGENT MOTIVATORS

Again, the agents are numbered according to their rank within the team, thus Agent #1 above is the leader and he is separated from the other two agents within his team by a very large enemy force. The values in Figure 8, above, reflect the friendly agents' motivations just prior to being killed by the large enemy force. From the chart it is clear that discipline is the dominant motivator value in all three agents with enemy present. Thus all agents remain in Good Soldier mode and attempt to reposition with their leader prior to engaging the enemy. It is important to note, as a reminder, that we do see values in excess of 100 for the discipline motivator. Again, as described in Chapter IV, this is due to the ten points added to ensure a clear dominating value. Further analysis is necessary in order to determine whether or not the discipline motivator should be modified to allow for more realistic behavior in the face of such an overwhelming enemy force.

c. Discipline

In Figure 9 below, distToLdr, one can see that this scenario is similar to the scenario in Figure 7. The difference is that in this scenario the leader is isolated from the remainder of the team; whereas, in the other scenarios, the lone agent is not the leader. The reason for this is to try to isolate the distance to leader component of the discipline motivator equation. With the leader separated from the rest of the team, all agents should immediately begin engaging the enemy until such time that the three agents on the right discover that the leader is separated from the team. Upon this realization, they should enter Good Soldier mode and reposition with the leader, but not at the expense of turning a front to the enemy.

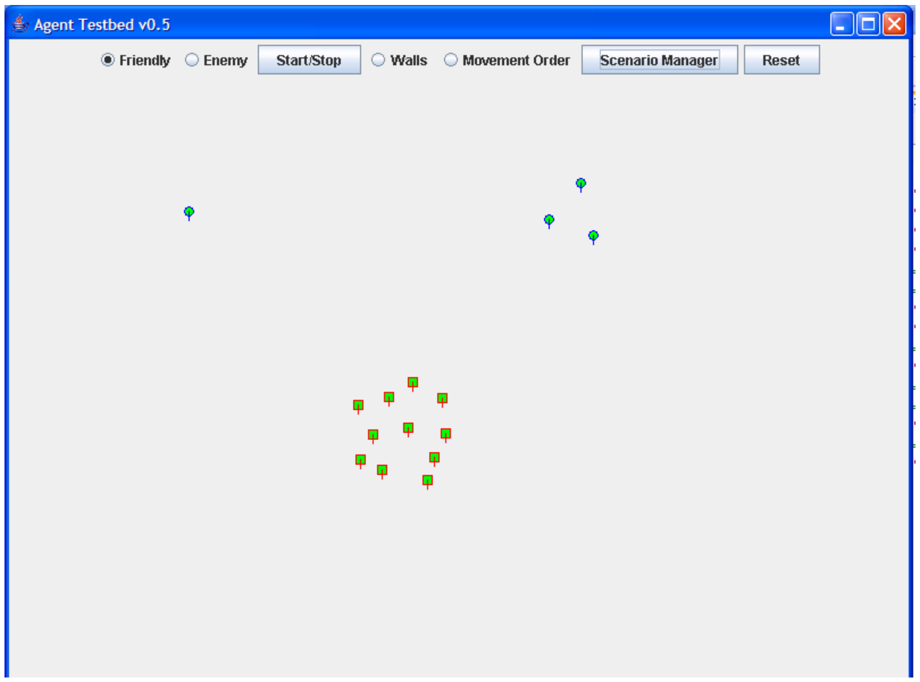


FIGURE 9. SCENARIO FILE: distToLdr.xml

In this scenario, with the enemy behaviors turned off, the discipline is not dynamic enough to allow for realistic friendly behaviors in the presence of the enemy. In fact the discipline value remains extremely high relative to the other motivators for the duration of the simulation run. The result is that the friendly agents remain in Good Soldier mode for the entire run and are not permitted to effectively engage the enemy. In fact, very often again the friendly agents move directly through the enemy in an effort to reposition into formation with the leader prior to engaging the enemy.

This scenario was initially created to test the discipline values for the respective agents when they are separated from their leader. In the process of conducting the analysis for discipline, some interesting points surfaced with regard to the insecurity and aggression values for the agents in this scenario. Specifically, when discipline and the enemy behaviors are off, the aggression values dominate placing the agents into Berserk mode until the entire enemy is dead, then the insecurity value dominates and the friendly agents go into a Search mode. There are at least two interesting points here when looking at the agents' resultant behaviors with regard to discipline. First, with discipline not in play, the agents do not attempt to re-position to take advantage of mass;

however, they do mass fires as a result of the aggression value, in lieu of repositioning. Second, they do not enter Scared mode and run away from the enemy. Thus, even with discipline set to zero, the resultant behavior of the agents still exhibits some form of discipline.

2. Verification of Motivator Equations

The following analysis was conducted using Microsoft Excel to verify the design of the motivator equations. The equation for each motivator was broken down into its individual components. The equations were entered into excel and graphs were generated to obtain visual representations of the motivator values over a spectrum of possible input variables.

a. Aggression

We initially evaluated the distance to target value of the aggression equation. This component of the aggression equation was designed to increase the aggression value of individual agents as the distance to the enemy decreases. When we graphed this value according to our initial equation, it was obvious that the inverse was true, that aggression actually decreased as agents moved closer to the enemy. Consequently, we adjusted the aggression motivator equation in the source code in order to obtain the desired resultant behavior. Below is a graph which verifies the design of the distance to target portion of the aggression equation after making the appropriate modifications to the equation. As the distance to the enemy increases, the distanceToTarget motivator value decreases.

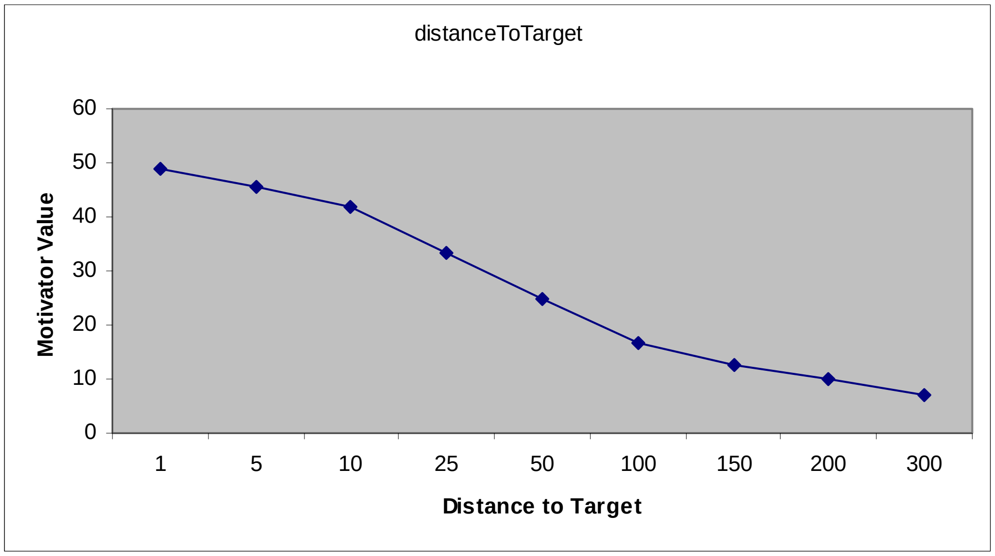


FIGURE 10. AGGRESSION EQUATION: Distance to Target Component

b. Discipline

The distance to leader component of the discipline equation was evaluated by graphing arbitrary, but possible values for the distance to the leader. The possible values for distToLdr value range from zero to thirty-four. It is clear from the graph that as soon as there is a separation from the leader, there is an immediate decrease in the agent's discipline value. From the graph, it is also clear that from zero to 200 meters separation from the leader, there is a much sharper decline in discipline than from 200 to 1000, where we see a much more gradual decline in the discipline value. While, admittedly, this was not by design, this is certainly a viable representation of discipline with respect to the distance from the leader. The majority of factors which might complicate discipline in such a small team will certainly manifest themselves as soon as there is a break with the leader. While there will continue to be confounding factors as distance from the leader increases, the additional factors are likely to be minimal when compared to the number introduced within the initial break with the leader.

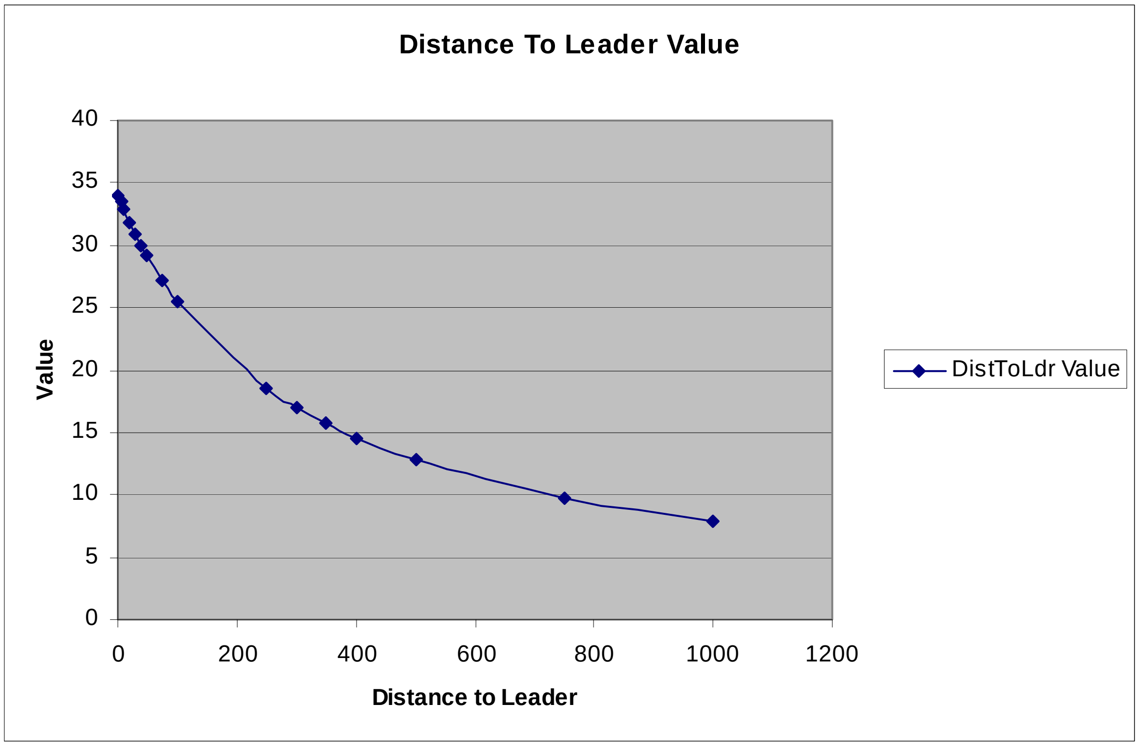


FIGURE 11. DISCIPLINE EQUATION: Distance to Leader Component

In Figure 12, below, the corresponding values for the last communications value are presented as a function of possible values of communication lapse. Communications lapse is presented here as time in seconds. This value should decrease as time increases; thus, modeling the decrease in discipline as communications breaks down or is interrupted. From the graph, the intended design of this component is well represented showing a gradual decaying affect on the discipline value over time.

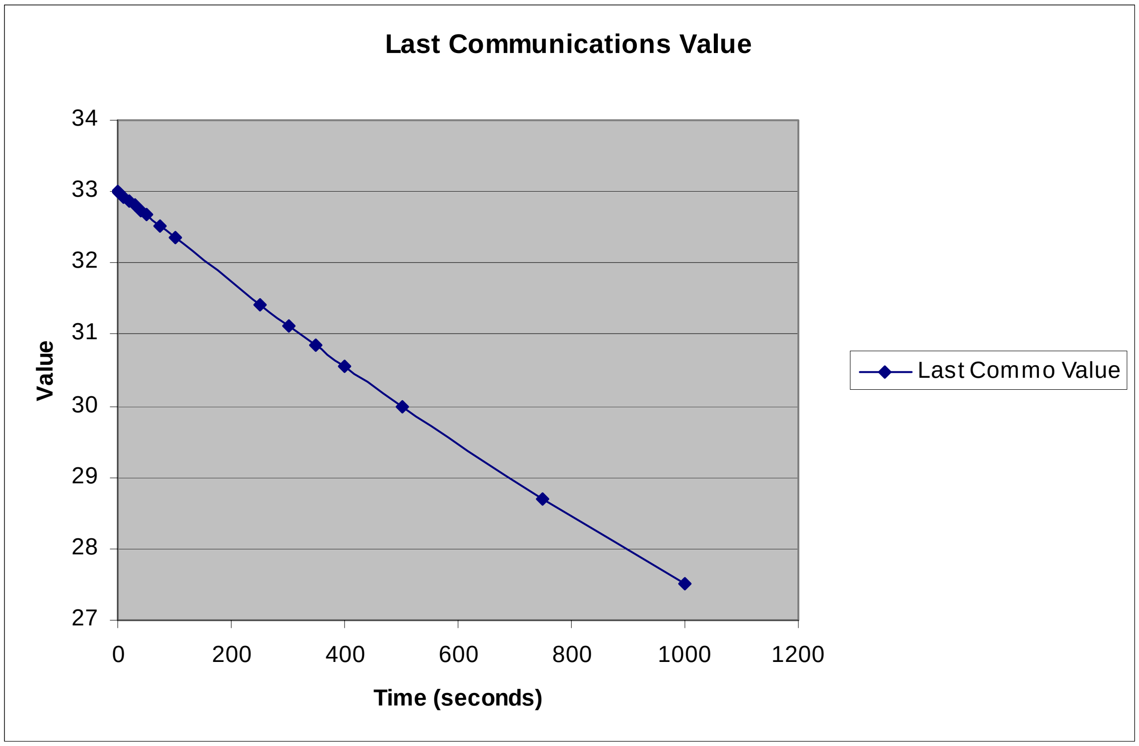


Figure 12. DISCIPLINE EQUATION: Last Communication Component

c. *Insecurity*

From the graph of the shots in vicinity component of the insecurity equation, as shots in the vicinity of agent increase, the resultant effect is an increase of the overall insecurity value. However, there is a clear problem with the results of this equation, since the majority of the increase in the shots in vicinity value occurs from zero to four shots in the vicinity. In this short range, the component value increases to twenty, almost topping out at the maximum for this value, twenty-five. As shots in vicinity increase beyond four out to seventy-five, the component value only increases five points versus twenty points from zero to four. This is a function of the expected number of shots each agent could expect to have in its vicinity. Currently, the equation is designed to reflect a baseline number of shots at twenty-five. Upon further analysis and observing successive runs of various scenarios, it was determined that this value was extremely high. Since the shots in vicinity value is calculated based upon a snapshot of shots in the vicinity of any one agent at any given time, it is almost impossible to actually encounter more than ten to fifteen shots in any given snapshot of the simulation engine. For this reason, we modified the modeled number of expected shots to reflect a maximum expected ten shots in the vicinity of any given agent.

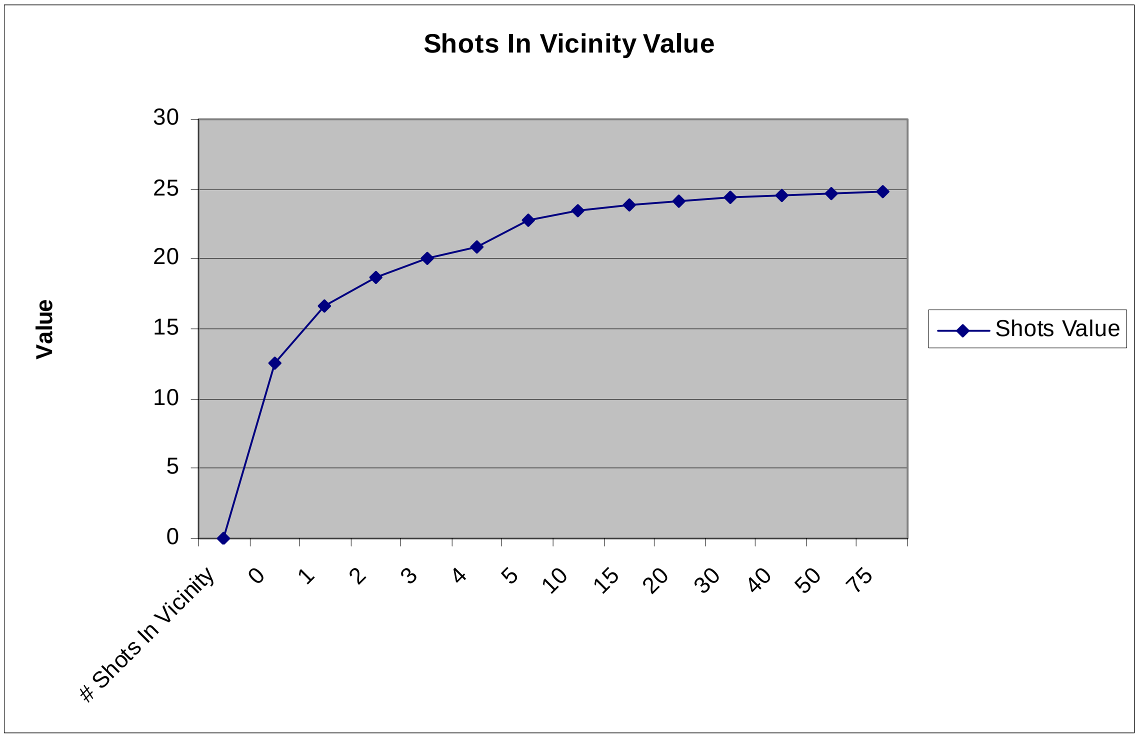


FIGURE 13. INSECURITY EQUATION: Shots In Vicinity Component

In Figure 14, we have graphed the distance to friendly center-of-mass component of the insecurity equation. Logically, as the distance from the friendly center-of-mass increases for an agent, the agent's insecurity should increase. This effect is modeled by our insecurity equation; however, there is also an unintended effect present. From the graph below, it is clear that this value will always be relatively high no matter how far an agent is separated from the friendly center-of-mass. This results in an undesirable over-weighting condition for this component of the insecurity equation.

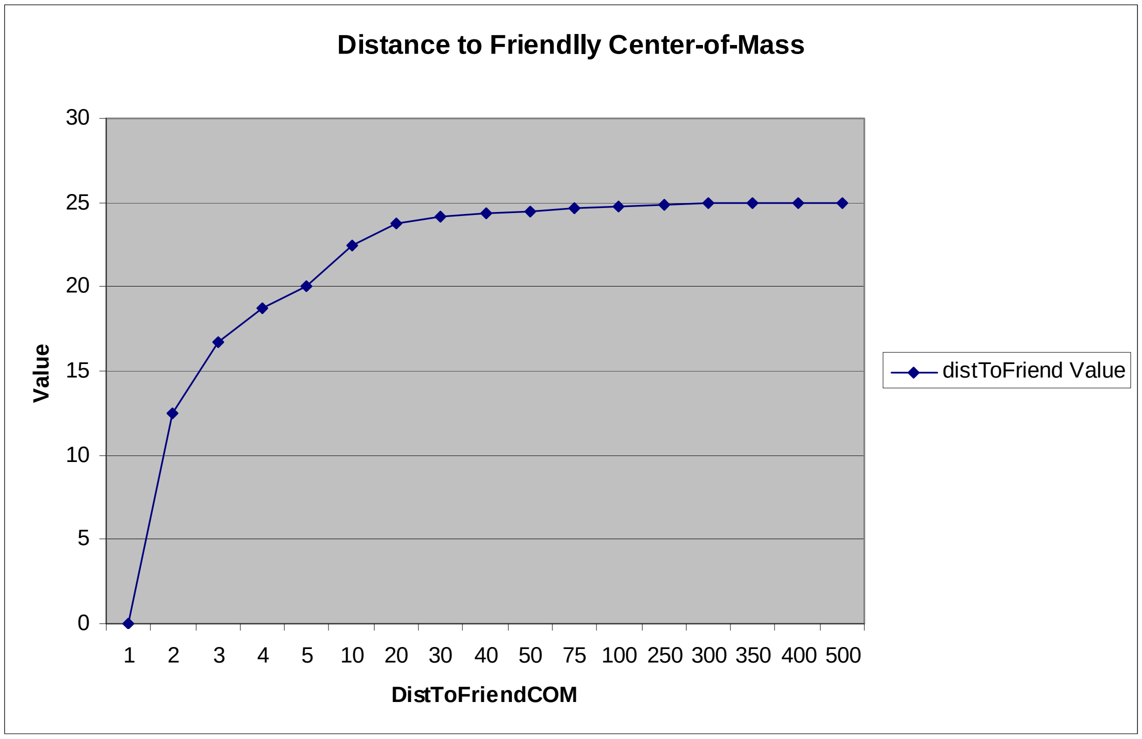


FIGURE 14. INSECURITY EQUATION: Distance to Friendly COM

3. Results of Motivator Analysis

a. Aggression

There were no significant changes to be made to the aggression equation. The only change is the correction to gain the desired effect of the decaying aggression the further an agent is from an enemy.

The aggression equation with the modified distanceToTarget component is presented below:

$$50 \times \frac{(\#Dead \times 2) + (\#Injured)}{\#Dead \times 2 + \#Injured + \#\,Alive + 1_{self}} + \left(50 \times \frac{OptimalDistance}{DistancetoTarget + OptimalDistance} \right)$$

b. *Discipline*

Throughout this analysis, the discipline motivator equation has continually been singled out as not dynamic enough to allow for realistic agent behaviors. The theme throughout was consistent, the discipline values were always too high to allow the agents to enter any other behavior mode. In an effort to make the discipline value more dynamic and to allow for more realistic agent behaviors in the presence of enemy, the discipline equation was restructured using a fraction of the individual components of the entire equation. Due to the drastic changes we made to the method of calculating discipline (in essence making all factors multiplicative in their effect) it is impractical to represent the calculation in an equation. Below is the portion of code which calculates the discipline value for the agent's behavior. The original code for the discipline equation is also included, but is commented out for comparison and reference.

```
//discipline, same incremental model
double newDiscipline = 0;
double discPercent;
if (teamLeader==idTag) //if the agent is the leader, give full credit
for health and distance
     discPercent = 1;    //newDiscipline += 67; // 67
else
  {
   //distance to leader
   double distanceToLeader =

distanceTo(ellipseToPoint(((Contact)contacts.get(teamLeader)).location)
);
   discPercent = 1 –

(distanceToLeader/(distanceToLeader+AgentRef.COMMO_RANGE))
//newDiscipline += (50 - (50 *

(distanceToLeader/(distanceToLeader+AgentRef.COMMO_RANGE))));
//leader health
if (((Contact)contacts.get(teamLeader)).strength==AgentRef.LIVE)


   discPercent = discPercent;
else if (((Contact)contacts.get(teamLeader)).strength!=AgentRef.DEAD)
    discPercent = discPercent * .5; //newDiscipline += 10;
  }
//time from last commo, max out at 10,000 ms (10 seconds) 33-
(33*(A2/(A2+5000)))
long commoLapse = Calendar.getInstance().getTimeInMillis() -
timeOfLastCommo;
discPercent = discPercent * (1 - (commoLapse/(commoLapse +
                       (AgentRef.COMMO_TIME*2))));
 //newDiscipline += (33 - (33*(commoLapse/(commoLapse +
                                 (AgentRef.COMMO_TIME*2)))));
newDiscipline = discPercent * 100;
discipline = (discipline * oldRatio) + (newDiscipline * newRatio);
```

c. *Insecurity*

In the equation, the distance to friendly center-of-mass component is modeled with a weighting equal to twenty-five percent of the overall insecurity value, equally weighted with the other three components of insecurity. In order to validly represent this component it was necessary to reduce the weighting from twenty-five percent to fifteen percent of the overall insecurity value. In addition to changing the weighting of the distance to friendly center-of-mass component, we also changed the weighting on the remaining three components to better reflect a more realistic situation in which a soldier might feel insecure. We set the shots in vicinity component weighting equal to the weighting of the live enemy ratio component, both were set to represent thirty-five percent each of the total insecurity value. We also reduced the injured agent component down from twenty-five to fifteen percent to match up with the weighting of the distance to friendly center-of-mass component.

These values more accurately represent the insecurity value which influences the behavior of our agents. The new insecurity equation follows:

$$(15\times \text{injured})+\left(35-\frac{10}{\text{shotsInVicinity}+1}\right)+\left(35\times\frac{\text{liveEnemy}}{\text{liveEnemy}+\text{liveFriendly}+1}\right)+\left(15-\frac{15}{\text{distanceTo}(\text{friendlyCOM})+1}\right)$$

C. TEAM-ENABLING BEHAVIORS

In this analysis, we will provide the reader with a more detailed explanation of how our agents interact and the role that the team-enabling behaviors play in creating the synergistic effects of teamwork described in our abstract model from Chapter III. Again, the team-enabling behaviors are: communications, synchronizing actions and team-member monitoring. In this section we will describe a single scenario and the agent behaviors as they relate to the team-enabling behaviors during a simulation run. Initially, all team-enabling behaviors will be active. This constitutes setting all three team-enabling behaviors to 1.0 within the GenericAgent class. This first run with all behaviors enabled will serve as the baseline run. We will then vary each behavior individually and report on the changes to the agent behaviors within the scenario. In this manner, the reader will gain some insight as to exactly how each team-enabling behavior affects the overall behaviors of the individual team members within the scenario. As a reminder from Chapter IV, when the agents are solid black, they are dead.

1. Scenario Description

In order to analyze the team-enabling behaviors, we designed a basic scenario in which a user could observe all of the team member behaviors and isolate those behaviors which are absent when the team-enabling behaviors are varied. To do this, we chose to go with a very basic scenario with enemy situated in the North and the friendly team in the South.

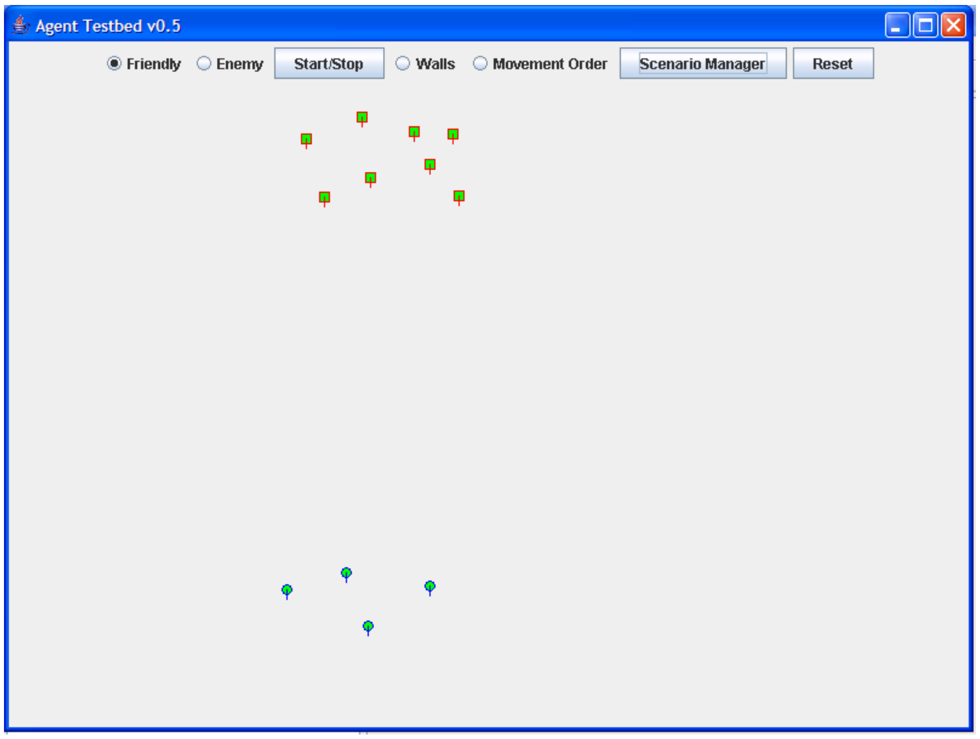


FIGURE 15. TEAM-ENABLING BEHAVIOR SCENARIO

For all runs in this portion of the analysis, the enemy agents are turned off. Thus the enemy will not shoot, move, or communicate. There presence and numbers alone provide the inputs to the motivator equations which drive the agent behaviors.

2. Baseline Run

In the baseline run for this descriptive analysis, we ran the simulation with all the team-enabling behaviors turned on. This run provides an idea of how the agents should act in this given scenario. When the simulation is started, the agents immediately begin communicating. They pass information about themselves and information about what they see, both friendly and enemy. This information is used effectively by all the agents given their respective ranks. The leader also passes orders to the other team members to reposition to the correct formation. We see in Figure 16 that the team has organized into a line formation so that they may effectively engage the enemy.

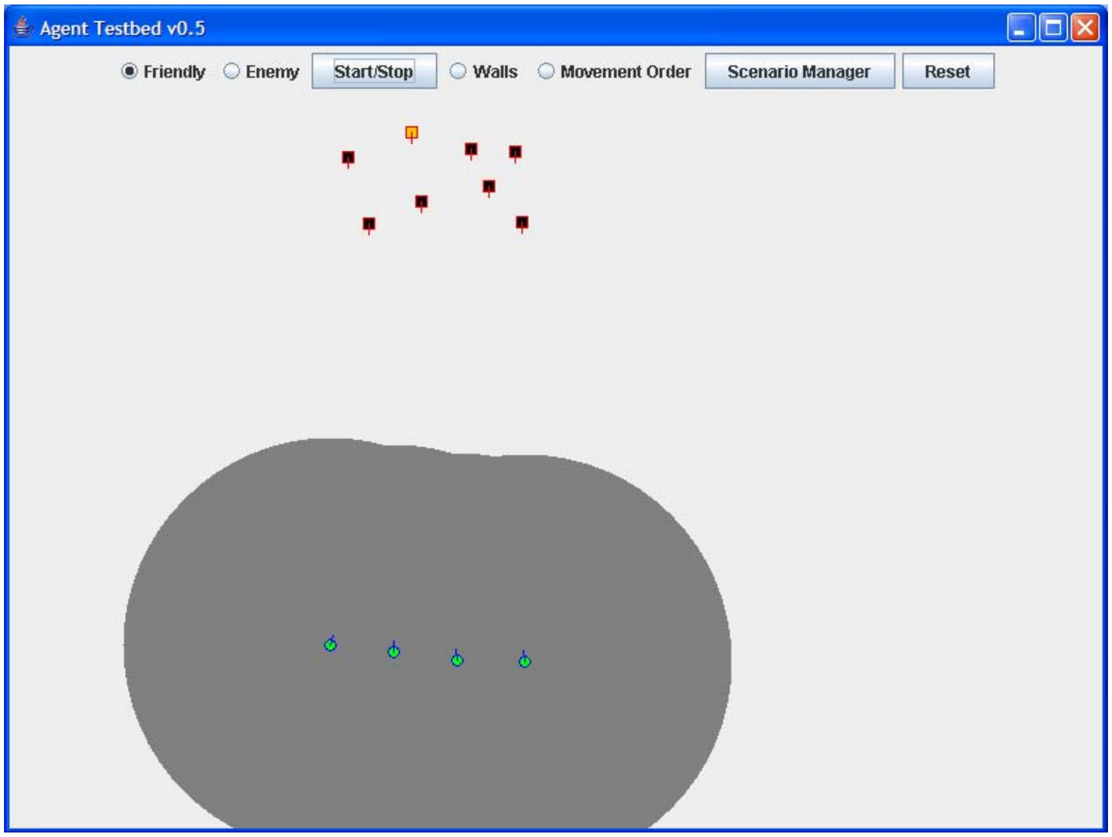


FIGURE 16. TEAM ON-LINE

As the team members move into their positions with in the line, other agents already in position check their fire to prevent fratricide. Once the team members reach their positions within the line formation, they start to engage the enemy. When the enemy is dead, this is effectively conveyed through communication to all agents. The leader then passes the message to return to coil formation and they re-enter Search mode once set in formation. Figure 17 shows the correct positions of the agents at the end of the run. All agents are spaced accordingly and are searching in the appropriate sectors.

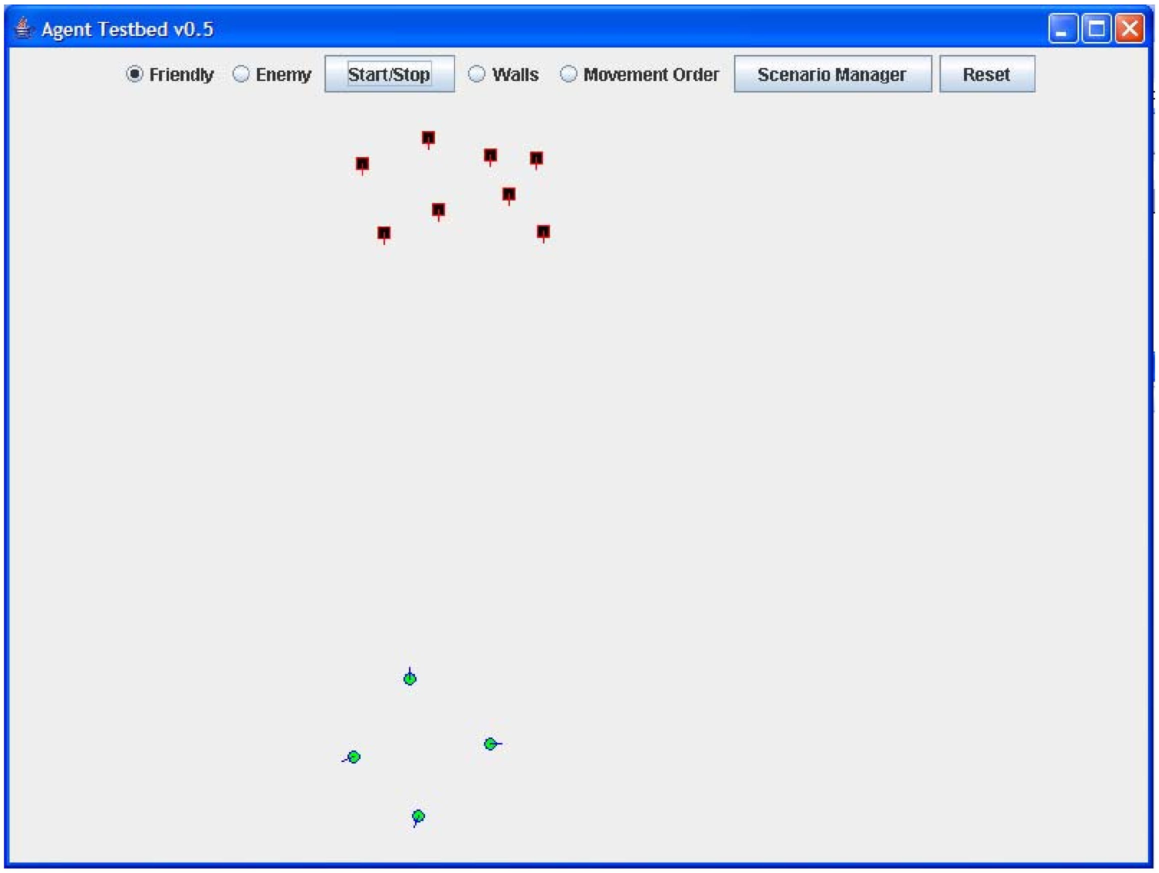


FIGURE 17. TEAM RETURNS TO COIL FORMATION

3. Varying Team-enabling Behaviors

a. Communication

For our first test of the effects of the team enabling behaviors, we "turned off" the agent's propensity to communicate. The agents only update their own situational awareness or contact list through their direct observation. As a result, when they conduct their initial searches of their surroundings, depending upon their locations, they visually identify each other and passively select agents as their leaders. However, there is a chance that in selecting a leader and subsequently moving to their default formation for that leader, they may have not seen all of their fellow friendly agents and therefore they might not select the same leader as everyone else. Their formation locations also may become misplaced as they miss important location updates from the messages sent out by their fellow agents.

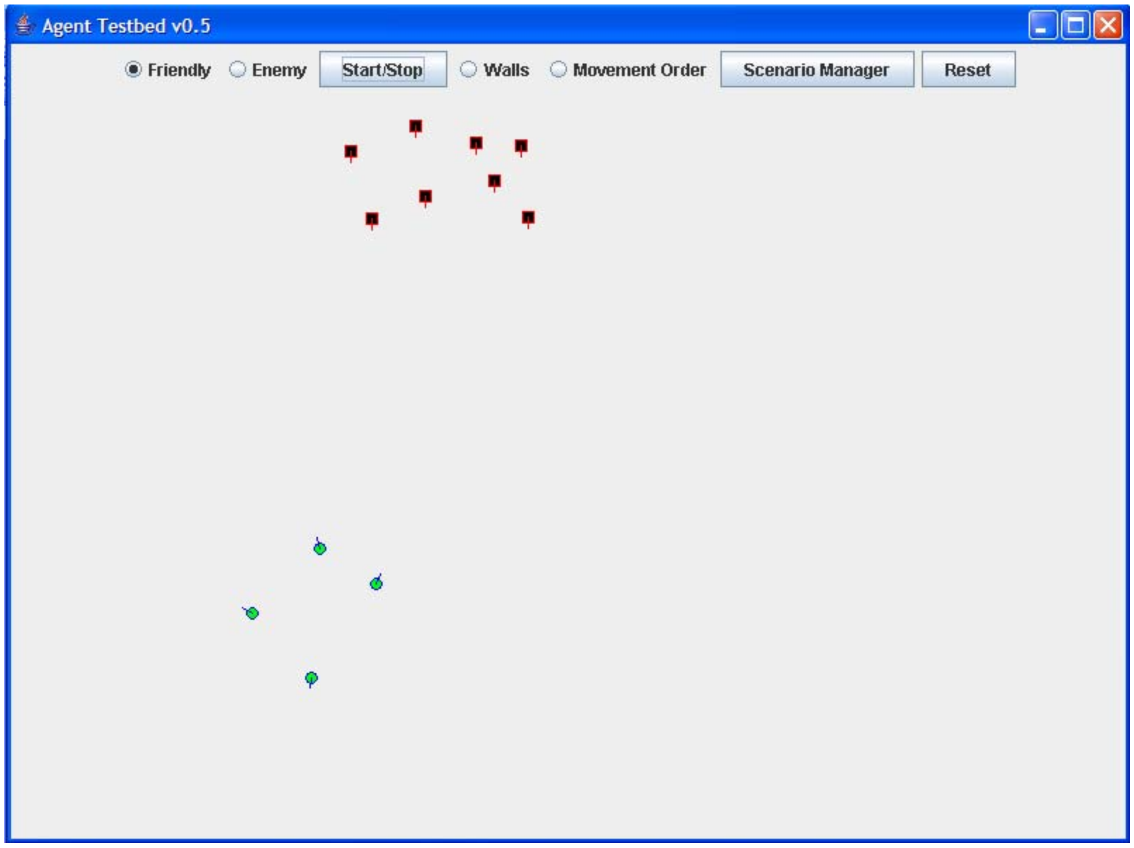


FIGURE 18. MISALIGNED COIL FORMATION

Because the agents default to coil formation, they will move into a coil formation once they recognize a leader. If, in the course of their movements, they directly observe enemy, once in formation, they will engage the enemy until the enemy they have personally observed is dead. Then they will resume their search patterns based on the last observed location of their peers. Additionally, since there is no communication, the leader never sends out the message to change formations to “line” in order to engage the enemy, or back to “coil” once the enemy is killed. We describe this behavior as passive or reactive teamwork. The agents don’t explicitly contribute to the “team”, they just react to it when they perceive it.

b. *Synchronizing Actions*

With a zero propensity to synchronize actions, the agents will communicate and observe each other and update their contact list appropriately. However, they will not adjust their orientation to account for the presence of their peers, and they will not move to take up their respective position in their designated formations. The agents have all the information necessary to do these actions, they just never use it. Therefore, they will simply attack any enemy while searching their surroundings and remain in their original locations.

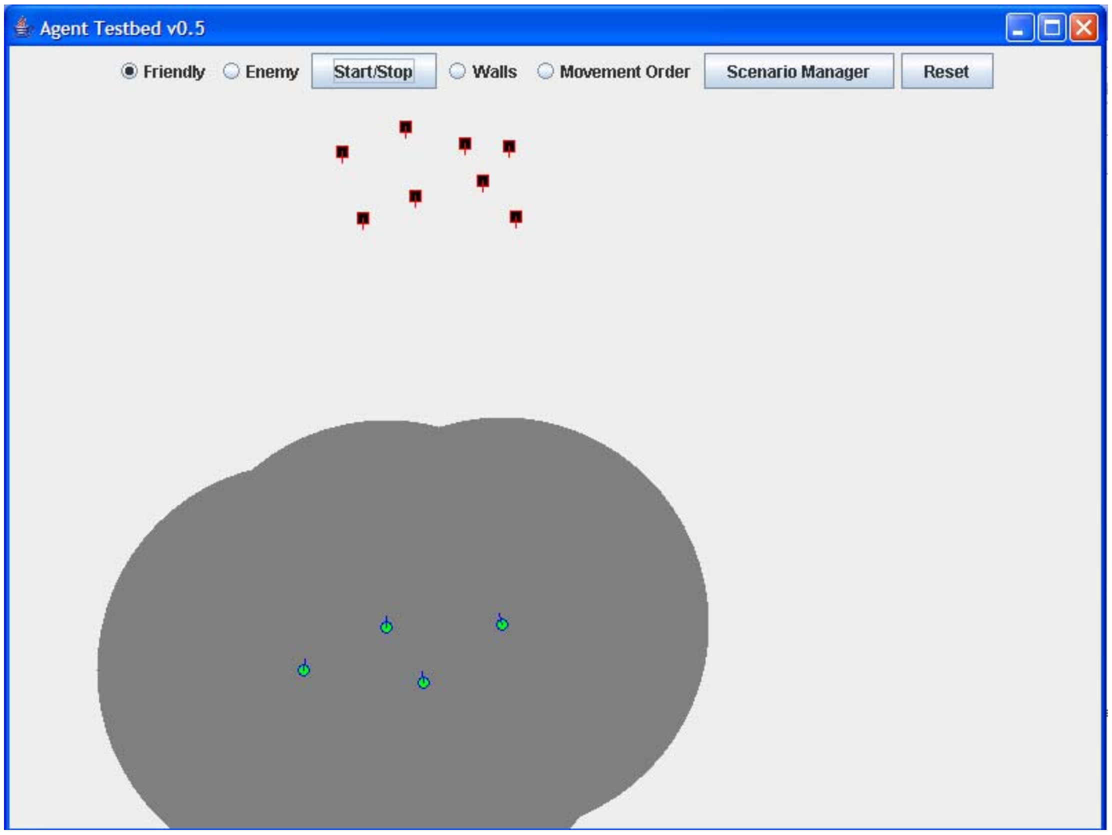


FIGURE 19. TEAM DOES NOT MOVE TO FORMATION

The agents in this test are cooperating in that they are sharing information. In this regard, they are interacting and contributing to the "team". However, the effects and appearance of any “teamwork” are diminished by their inability to move to and adjust their behavior based on the behavior and presence of their peers.

c. *Team-member Monitoring*

With no team member monitoring, the agent’s will disregard any communications from their peers, and will also disregard their own perception of their peers. As a result, they will never perceive another agent to be their leader, nor that they have any peers. Based on the communications protocol, they will not send any messages, nor will they move into any formations, since they see themselves as the team leaders of teams of one. Most detrimental to the team, however, is that they have not entered peer locations into their contact list, and when they check for fratricide there will never be a conflict. As a result they will fire upon an enemy even if a friendly agent is in the way.

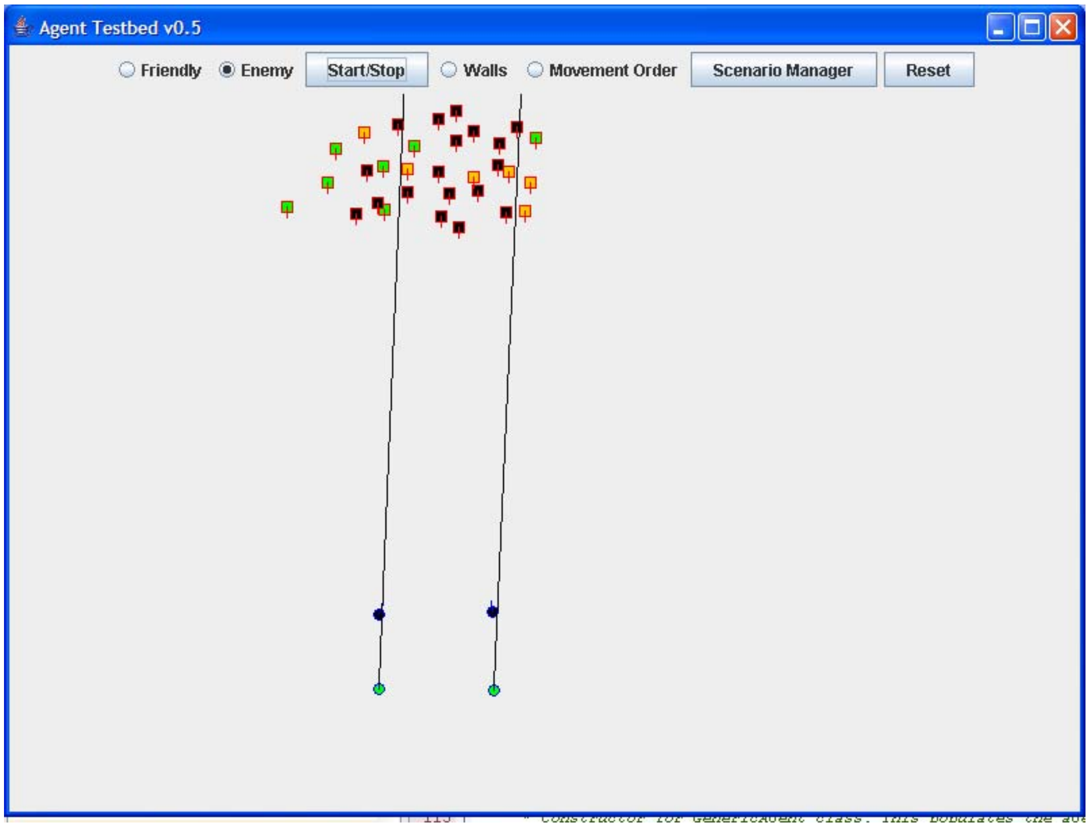


FIGURE 20. TEAM FRATRICIDE

In comparison with the above tests, this scenario seems to be the worst example of teamwork. The agents are unable to even recognize that they have teammates, even to the point of shooting them accidentally, and as a result there is no teamwork whatsoever.

VI. RESULTS AND CONCLUSIONS

Our thesis has presented an individual or agent based model for teamwork, as well as how this model might be implemented. The crux of our teamwork model is to be a springboard for future exploration of the individual's ability to perform its own Team Enabling Behaviors, described in our model as Team Member Monitoring, Synchronizing Actions, and Communications. Our simulation has demonstrated that, within our implementation, the lack of any one of those behaviors creates fully functional agents who can still operate in their environment, but who exhibit qualitatively diminished teamwork.

It should be noted that due to the bottom up nature of our work, we spent a large portion of our time creating a functional agent framework upon which to test our model. The aspects of the model, once tested, seem to verify the functional categorization of our theoretical model. However, the more challenging aspect of the work was in developing the underlying agents and simulations; hence the significant space in this thesis devoted to documenting our efforts. We believe it is important to note that we developed our agents with our model in mind. Though we tried to avoid skewing the agent performance in favor of our theory, the agents we designed had to be able to enact the behaviors we discussed.

It is our hope that this research might also serve as a springboard for future exploration of our theoretical teamwork model. We believe we have categorized the crucial behaviors needed for artificial intelligences to create the type of teamwork which we as 'real people' might take for granted. In the following chapter we list some ideas which we were unable to explore, but we believe might be beneficial future work on our model.

THIS PAGE INTENTIONALLY LEFT BLANK

VII. FUTURE RESEARCH

Throughout this thesis, we documented numerous opportunities to expand upon our work; here, we present several suggestions. This chapter can serve as a starting point for any further research on either enhancements to the abstract teamwork model or further development of the simulation created from the abstract model.

One of the first areas for future work would be to attempt to recreate our results in another simulation. Such work would serve to verify the validity of our theoretical model in a simulation which was not designed with the theory in mind. A couple simulations seem to lend themselves to this work. The US Army's ONESAF Objective Testbed provides tools which allow the user to define behaviors for the entities. ONESAF is mentioned above as an example of the "Teamwork" entity approach to synthetic teamwork, and as such, would provide a good baseline for comparison of teamwork approaches. The US Army also has a simulation named Infantry Warrior Simulation (IWARS), which is used to conduct high fidelity simulations of individual infantry. The individual infantry orientation of IWARS would provide a potential test bed for the implementation of our model.

There is also an opportunity to expand upon our model. One of our original suppositions was that teamwork should be most effective when applied within a hierarchical structure nesting echelons of teams with each "team leader" having between 3 and 5 subordinates. Future work may look at that aspect of the model and try to apply the same methodology for teamwork to a hierarchical structure. As it stands right now, the teams in our simulation contain all agents of the same side (all red or all blue agents) and are unbounded in their size. To achieve hierarchical structure, the agents would need to have a method do dynamically determine their team sizes and limits, as well as means of adjusting the teams as a result of casualties. They also need to be able to hierarchically organize themselves, and issue orders in proportion to their echelon. For example, a formation order from a squad leader will have implications over a vastly different distance from a formation order from a division commander.

A logical next step could be to create a scalable framework in which teamwork behaviors are repeatable. At each echelon above the very lowest, subordinates are themselves leaders within their own unit. The structure becomes a “team of teams”. Directives and goals as given by the highest leader are dissected and forwarded to each subordinate unit down the chain of command. At each level, the leader subdivides his assigned goals and parcels them out amongst his team. New issues of process management and scalability become apparent, but if successfully dealt with, the result would be simulated units that effectively aggregate their tasks. The results could be implemented at all levels of simulation application, and alleviate numerous problems associated with man-power requirements which effect simulation training. Further, such an aggregation could provide numerous opportunities to explore the macro-effects of leadership and decision making on large scale organizations.

As alluded to throughout the thesis, the simulation itself also offers a number of opportunities for expansion. We sought to maintain the balance between realism and simplicity when writing our code, and made a number of sacrifices in areas we would like to have spent more time refining.

Though walls are functional within the environment, the agents lack any navigational ability. This presents a number of interesting problems in terms of visibility and perception, and presents the question of whether to use a hidden navigational node structure or to somehow imbue the agents with an innate ability to determine the significance of different locations within the terrain. Even in the simplistic two dimensional environment of our simulation, this problem rapidly reached a level of complexity we were not yet willing to tackle. Along the same lines, the “movement order” user interface was never fully integrated. The framework is present, but the “movement order” data was never passed on to the agents to affect their behavior.

Another interesting path we did not tread was the concept of accentuating the cost of teamwork by preventing the agents from being able to talk and shoot at the same time. This might add a new dimension of realism to the agent behaviors and force some reconsideration of assumptions we made in our agents cognitive processes. Similarly, most communications were one-way. There is no expectation of acknowledgement of messages. It may be interesting to see if such awareness and prediction of agent behavior on the part of the agents themselves actually increases or reduces the amount of “chatter”.

As for our agents, there are many opportunities for changes there as well. Our solution to the agent motivators is just one of many possible solutions, and by no means do we believe ours to be the “ultimate” solution. Hopefully, we have presented a sound methodology for analyzing what might motivate an agent, but as this was not the focus of our research, there is plenty of room for refinement.

One area which stood out for us during the development of our agents, as mentioned several times throughout our thesis, is the definition of motivators which affect agents on a battlefield. We believe there is a potential for extensive research in an effort to quantify the impact of external events on the battlefield upon an agent’s goals, as well as the internal differences which make soldiers act in different ways when presented with the same perceptions.

The task of analyzing the result of runs could also be simplified by incorporating more robust statistical collection. An example of an API which does this very well is the statistics collection capabilities of SimKit and VisKit, an application designed by Professor Arnie Buss at the Naval Postgraduate School.

Last, but certainly not least, is the issue of emergent behavior versus our own implementation, which required a significant amount of data to represent the agent’s environment. This technique worked well for us when exploring the interactions of a small team of three to five soldiers. However, as the number of agent’s agents and complexity of the environment increase, the cost of having a detailed world representation within each agent may become prohibitive. In order to increase efficiency of our model at larger scales, it would be very beneficial to explore optimizations for our code, as well as exploring the grey area between emergent agents and those having a detailed, expensive world representation.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Burgess, Rene G., and Christian Darken (2004). *Realistic Human Path Planning using Fluid Simulation.* Monterey: Naval Postgraduate School.

Darken, Christian J. (2005). *Toward Learned Anticipation in Complex Stochastic Environments.* Monterey: Naval Postgraduate School.

Darken, Christian J., David J. Morgan, and Gregory Paull (2004). *Efficient and Dynamic Response to Fire.* Monterey: Naval Postgraduate School.

Darken, Christian J., Gregory Paull (2004). *Integrated On- and Off-line Cover Finding and Exploitation.* Monterey: Naval Postgraduate School.

Darken, Christian J (2004). *Visibility and Concealment Algorithms for 3D Simulations.* Monterey: Naval Postgraduate School.

Endsley, Mica R. and Daniel J. Garland (2000). *Situational Awareness Analysis and Measurement.* Mahwah: Lawrence Erlbaum Associates.

Eccles, D.W. and Gershon Tenenbaum (2004). Why an Expert Team is More Than a Team of Experts: A Social-Cognitive Conceptualization of Team Coordination and Communication in Sport. *Journal of Sport and Exercise Psychology, 26*, 542-560.

Fiore, S and E. Salas (2004). *Team Cognition.* Washington, DC: American Psychological Association.

Harvard Business Review (2001). *On Decision Making.* Boston: Harvard Business School Publishing.

Hastie, Reid and Robyn M. Dawes (2001). *Rational Choice In An Uncertain World.* Thousand Oaks: Sage Publications.

Henderson, Wm. Darryl (1985). *Cohesion: The Human Element in Combat.* Washington DC: National Defense University Press.

Holland, J. (1998). *Emergence.* Reading: Perseus Books.

Joint Publication 3-06 (2002). *Doctrine For Joint Urban Operations.* Suffolk : United States Joint Forces Command, Joint Warfighting Center.

Kauffman, S. (1994). *At Home in the Universe.* New York: Oxford University Press.

Kendall, D. and E. Salas (2004). Measuring Team Performance: Review of Current Methods and Consideration of Future Needs. In J.W. Ness, V. Tepe, and D.R. Ritzer (Eds.), *The Science and Simulation of Human Performance, Volume 5: Advances in Human Performance and Cognitive Engineering Research* (307-326). Netherlands: Elsevier, Inc.

Klein, Gary (1999). *Sources of Power.* Boston: The MIT Press.

Koehler, M. (2002). Emergence and the Military, Is It Important or Strictly for the Boids? In G. Horne & S. Johnson (Eds.), *Maneuver Warfare Science 2002* (pp.149-179). Quantico: USMC Project Albert.

Krulak, C. (1999). The Strategic Corporal: Leadership in the Three Block War. In *Marines Magazine*. Quantico: U.S. Marine Corps.

Larimer, Larry (2004). *Soldier Modeling and Analysis Working Group (MAWG) Evaluation Report.* White Sands Missile Range: TRADOC Analysis Center-White Sands Missile Range (TRAC-WSMR).

Marshall, S.L.A. (1978). *Men Against Fire.* Gloucester: Peter Smith.

Mavor, Anne S. and Richard W. Pew (1998). *Modeling Human and Organizational Behavior.* Washington: National Academy Press.

Orkin, Jeff (2003). Simple Techniques for Coordinated Behavior. In Steve Rabin (Ed.), *AI Game Programming Wisdom 2* (pp. 199-206). Hingham, Massachusetts: Charles River Media, Inc.

Popper, Micha (1996). Leadership in Military Combat Units and Business organizations. *Journal of Managerial Psychology, 11*, 15-23.

Rabin, S. (Ed.). (2002). *AI Game Programming Wisdom.* Hingham: Charles River Media.

Rabin, S. (Ed.). (2004). *AI Game Programming Wisdom 2.* Hingham: Charles River Media.

Reynolds, John (2002). Tactical Team AI Using a Command Hierarchy. In Steve Rabin (Ed.), *AI Game Programming Wisdom* (pp. 260-271). Hingham, Massachusetts: Charles River Media.

Sarter, Nadine B., and David D. Woods (1991). Situation Awareness: A Critical But Ill-Defined Phenomenon. *The International Journal of Aviation Psychology*. Hillsdale: Lawrence Erlbaum Associates, Inc.

Sorkin, Robert D. (2002) *Assessing and Improving Team Decision Making.* Orlando: University of Florida.

Stewart, G.L., C.C. Manz, and H.P. Sims, Jr. (1999). *Teamwork and Group Dynamics.* New York: John Wiley & Sons, Inc.

Therrien, Sakura S. (2002). *A Bayesian Model to Incorporate Human Factors in Commander's Decision Making.* Monterey: Naval Postgraduate School.

van der Sterren, W. (2002). Squad Tactics: Team AI and Emergent Maneuvers. In Steve Rabin (Ed.), *AI Game Programming Wisdom* (pp. 233-246). Hingham: Charles River Media.

van der Sterren, W. (2002). Squad Tactics: Planned Maneuvers. In Steve Rabin (Ed.), *AI Game Programming Wisdom* (pp. 247-259). Hingham: Charles River Media.

Zaccaro, Klimoski (2002). The Interface of Leadership and Team Processes. *Group & Organization Management*, 27 (pp. 4-13).

Zsambok, Caroline E. and Gary Klein (1997). *Naturalistic Decision Making.* Mahwah: Lawrence Erlbaum Associates.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Leroy A. Jackson
   Deputy Director, Training and Analysis Command-Monterey
   Monterey, California

4. Rodney Barber
   Directorate of Battle Command, Simulation and Experimentation
   Crystal City, Virginia

5. Curt Blais
   MOVES Institute
   Monterey, California